M 11

```
DDDDDDDDDDDD        CCCCCCCCCCCC  LLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
```

```
**FILE**ID**READREC                                              G 7




RRRRRRRR    EEEEEEEEEE    AAAAAA    DDDDDDDD    RRRRRRRR    EEEEEEEEEE    CCCCCCCC
RRRRRRRR    EEEEEEEEEE    AAAAAA    DDDDDDDD    RRRRRRRR    EEEEEEEEEE    CCCCCCCC
RR      RR  EE          AA      AA  DD      DD  RR      RR  EE          CC
RR      RR  EE          AA      AA  DD      DD  RR      RR  EE          CC
RR      RR  EE          AA      AA  DD      DD  RR      RR  EE          CC
RR      RR  EE          AA      AA  DD      DD  RR      RR  EE          CC
RRRRRRRR    EEEEEEEE    AA      AA  DD      DD  RRRRRRRR    EEEEEEEE    CC
RRRRRRRR    EEEEEEEE    AA      AA  DD      DD  RRRRRRRR    EEEEEEEE    CC
RR  RR      EE          AAAAAAAAAA  DD      DD  RR  RR      EE          CC
RR  RR      EE          AAAAAAAAAA  DD      DD  RR  RR      EE          CC
RR    RR    EE          AA      AA  DD      DD  RR    RR    EE          CC              ....
RR    RR    EE          AA      AA  DD      DD  RR    RR    EE          CC              ....
RR      RR  EEEEEEEEEE  AA      AA  DDDDDDDD    RR      RR  EEEEEEEEEE    CCCCCCCC      ....
RR      RR  EEEEEEEEEE  AA      AA  DDDDDDDD    RR      RR  EEEEEEEEEE    CCCCCCCC      ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

Table of contents

```
0000            .TITLE  READREC - READ AN INPUT RECORD
0000            .IDENT  'V04-000'
0000            .DEFAULT DISPLACEMENT,WORD
0000
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
0000    ; ABSTRACT:
0000    ;
0000    ;       These routines are called by the lexical processing routines to
0000    ;       perform functions which are optional to the basic lexical routines,
0000    ;       but are required by CLI parsing.
0000    ;
0000    ; AUTHOR:
0000    ;
0000    ;       Tim Halvorsen, Jan-1982
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;       V03-021 HWS0092         Harold Schultz  22-Jul-1984
0000    ;               Add support for execute-only command procedures.
0000    ;
0000    ;       V03-020 HWS0077         Harold Schultz  29-Jun-9184
0000    ;               If EOF encountered when reading a command procedure
0000    ;               while processing a line continuation, ignore termination
0000    ;               and return EOL.
0000    ;
0000    ;       V03-019 HWS0072         Harold Schultz  08-Jun-1984
0000    ;               Partially undo HWS0006 (setting of the parse position).
0000    ;               Set the parse position prior to processing the @FILESPEC
0000    ;               construct, but save the old parse position set by the
0000    ;               higher level routines and restore it after the '@' processing
0000    ;               has successfully completed.
0000    ;
0000    ;       V03-018 HWS0039         Harold Schultz  26-Mar-1984
0000    ;               When using arrow keys, don't duplicate command when changing
```

```
0000    ;               directions in recalling commands and are at the beginning of
0000    ;               the buffer.
0000    ;
0000    ;       V03-017 HWS0034         Harold Schultz  16-Mar-1984
0000    ;               Always use terminal RAB when outputting escape sequences
0000    ;               and arrow keys.
0000    ;
0000    ;       V03-016 HWS0006         Harold Schultz  13-Feb-1984
0000    ;               Remove parse location setting (DCL$MARK) when
0000    ;               processing an indirect command file. Let higher
0000    ;               level routines set parse location.
0000    ;               Use PRC_V_CARRCNTL to determine whether or not
0000    ;               a CR/LF is to be inserted before the prompt string.
0000    ;               Fix RECALL_NEXT so it will not skip first command
0000    ;               at bottom of command buffer when recall buffer
0000    ;               not full.
0000    ;               Save RMS STS and STV values in PRC data table
0000    ;
0000    ;       V03-015 PCG0017         Peter George    03-Jan-1984
0000    ;               Move location of test for PRC_V_FLUSH.
0000    ;
0000    ;       V03-014 PCG0017         Peter George    03-Jan-1984
0000    ;               Use PRC_V_FLUSH to deterime how to handle EOF when
0000    ;               peforming a flush.
0000    ;
0000    ;       V03-013 PCG0016         Peter George    18-Nov-1983
0000    ;               Support up and down arrow recall.
0000    ;               Do not automatically close command procedures when
0000    ;               flushing a record.  Add support for erase keypad attribute.
0000    ;
0000    ;       V03-012 PCG0015         Peter George    27-Sep-1983
0000    ;               Ignore spurious CTRL/Y's.
0000    ;               Only check for CTRL/B's and ESC if input is from terminal.
0000    ;
0000    ;       V03-011 PCG0014         Peter George    17-Jul-1983
0000    ;               Support 20 recalled commands.
0000    ;
0000    ;       V03-010 PCG0013         Peter George    01-May-1983
0000    ;               Correctly signal keypad buffer overflows.
0000    ;               Fix CTRL/Y interrupting GOTO bug.
0000    ;
0000    ;       V03-009 PCG0012         Peter George    20-Apr-1983
0000    ;               Check for CTRL/B with CMPB.
0000    ;
0000    ;       V03-008 PCG0011         Peter George    06-Apr-1983
0000    ;               Remove GOTO code.
0000    ;               Reformat DCL$INPUT code.
0000    ;               Change test for escape sequence.
0000    ;               Allow the RECALL command to accept letters as arguments.
0000    ;               Update recall buffer when at ctrl/y level.
0000    ;
0000    ;       V03-007 PCG0010         Peter George    01-Apr-1983
0000    ;               Change STV MOVZWL to MOVZBL.
0000    ;
0000    ;       V03-006 PCG0009         Peter George    24-Feb-1983
0000    ;               Do not verify lines in EXE-only command procedures.
0000    ;               Lookup terminating escape sequences in the keypad
```

```
0000   ;                  symbol table and act on them.
0000   ;                  Use new XABTRM and init appropriate fields in its item list.
0000   ;                  Add RECALL command and CTRL/B processing.
0000   ;
0000   ;          V03-005 PCG0008         Peter George    10-Feb-1983
0000   ;                  Close SYS$OUTPUT on silent logout.
0000   ;
0000   ;          V03-004 PCG0007         Peter George    15-Jan-1983
0000   ;                  Supply more rigourous test of whether commands
0000   ;                  should be verified.  Close PPF files before silent logout.
0000   ;
0000   ;          V03-003 PCG0006         Peter George    28-Dec-1982
0000   ;                  If $GET fails because of insufficient quota,
0000   ;                  then log the process out.
0000   ;
0000   ;          V03-002 PCG0005         Peter George    14-Nov-1982
0000   ;                  Call DCL$UPCASE instead of DCL$REMBLANKS
0000   ;                  Use prompt descriptor instead of WRK_L_PROMPT.
0000   ;
0000   ;          V03-001 PCG0004         Peter George    28-Oct-1982
0000   ;                  Get prompt string from PRC.
0000   ;                  Process escape sequences.
0000   ;---
```

```
                0000  ;
                0000  ; MACRO LIBRARY CALLS
                0000  ;
                0000
                0000            PRCDEF                          ;DEFINE PROCESS WORK AREA
                0000            WRKDEF                          ;DEFINE COMMAND WORK AREA
                0000            ITRMDEF                         ;XABTRM ITEM LIST DEFINITIONS
                0000            SYMDEF                          ;DEFINE SYMBOL TABLE ENTRY FORMAT
                0000            PTRDEF                          ;DEFINE TOKEN DESCRIPTORS
                0000            $CLIMSGDEF                      ;DEFINE ERROR/STATUS VALUES
                0000            $FABDEF                         ;DEFINE FAB OFFSETS
                0000            $RABDEF                         ;DEFINE RAB OFFSETS
                0000            $XABTRMDEF                      ;DEFINE TERMINAL XAB
                0000            $DVIDEF                         ;DEFINE $GETDVI ITEM CODES
                0000            $TT2DEF                         ;DEFINE DEVDEPEND2 BITS
                0000
            00000000            .PSECT  DCL$ZCODE,BYTE,RD,NOWRT
                0000
          04    0000  ERASE:    .BYTE   4
    4B5B1B0D    0001            .LONG   ^X4B5B1B0D              ;<CR>ESC[K
```

```
                        0005          .SBTTL  READ NEXT INPUT RECORD
                        0005  ;+
                        0005  ; DCL$INPUT - READ NEXT INPUT RECORD
                        0005  ;
                        0005  ; READS THE NEXT INPUT RECORD AND PLACES IT INTO THE INPUT BUFFER.
                        0005  ; THE CHARACTER POINTER IS RESET TO THE BEGINNING OF THE RECORD.  A
                        0005  ; SYMBOL SUBSTITUTION PASS IS PERFORMED IF A SINGLE OCCURRENCE OF A
                        0005  ; SINGLE QUOTE IS DETECTED IN THE RECORD.
                        0005  ;
                        0005  ; INPUTS:
                        0005  ;
                        0005  ;       R11 = ADDRESS OF PRC AREA
                        0005  ;       R10 = ADDRESS OF WRK AREA
                        0005  ;
                        0005  ; OUTPUTS:
                        0005  ;
                        0005  ;       WRK_L_CHARPTR POINTS TO BEGINNING OF INPUT RECORD, WHICH
                        0005  ;               HAS BEEN TERMINATED BY A NULL BYTE.
                        0005  ;
                        0005  ;       R0 = FIRST CHARACTER IN INPUT BUFFER
                        0005  ;-
                        0005  DCL$INPUT::                                             ;INPUT NEXT RECORD
              1C   BB   0005          PUSHR   #^M<R2,R3,R4>                           ;SAVE REGISTERS
                        0007
                        0007  ;
                        0007  ; IF AUTOLOGO FLAG SET AND WE ARE AT LEVEL 0 OR CTRL/Y LEVEL,
                        0007  ; THEN DELETE THIS PROCESS TO PERFORM AN IMPLIED LOGOUT BUT WITHOUT
                        0007  ; ANY LOGOUT MESSAGE.  THIS IS FOR THE SPAWN COMMAND.
                        0007  ;
 12 68 AB     08   E1   0007  REINP:  BBC     #PRC_V_AUTOLOGO,PRC_W_FLAGS(R11),20$    ;BRANCH IF FLAG NOT SET
 0D F0 AA     07   E0   000C          BBS     #WRK_V_INQUIRE,WRK_W_FLAGS(R10),20$     ;BRANCH IF INQUIRE
 05 68 AB     0B   E0   0011          BBS     #PRC_V_YLEVEL,PRC_W_FLAGS(R11),10$      ;IF SET, AT CONTROL Y/C LEVE
           5C AB   D5   0016          TSTL    PRC_L_INDEPTH(R11)                      ;INDIRECT LEVEL ZERO?
              03   12   0019          BNEQ    20$                                     ;BRANCH IF NOT
            027D   31   001B  10$:    BRW     SILENT_LOGOUT                           ;PERFORM SILENT LOGOUT
                        001E
                        001E  ;
                        001E  ; GET ADDRESS OF THE RAB ASSOCIATED WITH THIS INDIRECT LEVEL
                        001E  ;
 0E F0 AA     07   E0   001E  20$:    BBS     #WRK_V_INQUIRE,WRK_W_FLAGS(R10),30$     ;IF SET, QUERY IN PROGRESS
    54     14 AB   D0   0023          MOVL    PRC_L_INDINPRAB(R11),R4                 ;GET ADDRESS OF LEVEL N RAB
 09 68 AB     0B   E1   0027          BBC     #PRC_V_YLEVEL,PRC_W_FLAGS(R11),40$      ;IF CLR, NOT AT CONTROL Y/C
 04 68 AB     04   E0   002C          BBS     #PRC_V_GOTO,PRC_W_FLAGS(R11),40$        ;IF SET, AT GOTO LEVEL
    54     08 AB   D0   0031  30$:    MOVL    PRC_L_INPRAB(R11),R4                    ;GET ADDRESS OF LEVEL 0 RAB
                        0035
                        0035  ;
                        0035  ; SETUP PROMPT STRING
                        0035  ;
       50  10 AB   D0   0035  40$:    MOVL    PRC_L_TRMLIST(R11),R0                   ;GET ADDRESS OF XABTRM ITEM
 0C A0   F99E CA   B0   0039          MOVW    WRK_W_PMPTLEN(R10),ITRM_W_PMPTLEN(R0)   ;SET LENGTH OF PROMPT STRING
 10 A0   F9A2 CA   D0   003F          MOVL    WRK_L_PMPTADDR(R10),ITRM_L_PMPTADDR(R0) ;SET ADDRESS OF PROMPT STRIN
                        0045
```

```
                                    0045   213 ;
                                    0045   214 ; SETUP INPUT BUFFER AND POINTERS.
                                    0045   215 ;
                                    0045   216 GET_INPUT:
      20 A4     0100 8F       B0    0045   217          MOVW     #WRK_C_INPBUFSIZ,RAB$W_USZ(R4)          ;SET SIZE OF INPUT BUFFER
         52     F894 CA       9E    004B   218          MOVAB    WRK_G_INPBUF-2(R10),R2                  ;GET ADDRESS OF INPUT BUFFER
            24 A4     52      D0    0050   219          MOVL     R2,RAB$L_UBF(R4)                        ;SET ADDRESS OF INPUT BUFFER
   F4BE CA      FF A2         9E    0054   220          MOVAB    -1(R2),WRK_L_CHARPTR(R10)               ;SET POINTER FOR GET CHARACT
                                    005A   221
                                    005A   222 ;
                                    005A   223 ; READ THE NEXT INPUT RECORD AND CHECK FOR ERRORS.
                                    005A   224 ;
                                    005A   225          DISABLE                                          ;DISABLE CONTROL Y/C AST'S
        05 F0 AA      07      E1    005E   226          BBC      #WRK_V_INQUIRE,WRK_W_FLAGS(R10),10$     ;SKIP IF NOT INQUIRING
                                    0063   227          SETBIT   RAB$V_PPF_IND,RAB$W_ISI(R4)             ;SET INDIRECT PPF
                                    0068   228 10$:     $GET     RAB=(R4)                                ;GET NEXT RECORD FROM INPUT
                                    0071   229          CLRBIT   RAB$V_PPF_IND,RAB$W_ISI(R4)             ;CLEAR INDIRECT PPF
        23 68 AB      01      E1    0076   230          BBC      #PRC_V_CNTRLY,PRC_W_FLAGS(R11),20$      ;BRANCH IF NO CTRL/Y PENDING
  1A 18 A4   00000000'8F      E1    007B   231          BBC      #DEV$V_TRM,RAB$L_CTX(R4),20$            ;SKIP IF NOT TERMINAL
       00000000'8F    50      D1    0084   232          CMPL     R0,#RMS$_CONTROLY                       ;DOUBLE CHECK FOR WINDOW
                      0D      12    008B   233          BNEQ     15$                                     ;SPURIOUS CTRL/Y
            51     40 A4      D0    008D   234          MOVL     RAB$L_XAB(R4),R1                        ;GET ADDRESS OF XABTRM
                      18      D0    0091   235          MOVL     #ITRM_K_MINLEN,-                        ;SET SHORT LENGTH OF ITEM LI
                   0C A1            0093   236                   XAB$W_ITMLST_LEN(R1)                    ;
                    0519      30    0095   237          BSBW     DCL$LOCKED_STATE                        ;RESTORE LOCKED STATE
                      04      11    0098   238          BRB      20$                                     ;
                                    009A   239 15$:     CLRBIT   PRC_V_CNTRLY,PRC_W_FLAGS(R11)           ;IGNORE THIS CTRL/Y
                                    009E   240 20$:     ENABLE                                           ;ENABLE CONTROL Y/C AST'S
                   0E 50      E8    00A0   241          BLBS     R0,PROCESS_INPUT                        ;IF LBC I/O ERROR
           0000'8F    50      B1    00A3   242          CMPW     R0,#RMS$_SYS&^XFFFF                     ;ERROR IN QIO?
                      04      12    00A8   243          BNEQ     ERROR1                                  ;NO, THEN SKIP
            50     0C A4      D0    00AA   244          MOVL     RAB$L_STV(R4),R0                        ;GET PARTICULAR ERROR STATUS
                    00EA      31    00AE   245 ERROR1:  BRW      IO_ERROR                                ;ERROR
                                    00B1   246
```

```
                          00B1   ;
                          00B1   ; READ COMPLETED SUCCESSFULLY.  CLEAN UP AND THEN PROCESS THE INPUT RECORD.
                          00B1   ;
                          00B1   PROCESS_INPUT:
        53    22 A4    3C 00B1           MOVZWL  RAB$W_RSZ(R4),R3                     ;GET LENGTH OF INPUT RECORD
        51    40 A4    D0 00B5           MOVL    RAB$L_XAB(R4),R1                     ;GET ADDRESS OF XABTRM
                 18    D0 00B9           MOVL    #ITRM_K_MINLEN,-                     ;SET SHORT LENGTH OF ITEM LI
            0C A1         00BB                   XAB$W_ITMLST_LEN(R1)                 ;
        50    10 AB    D0 00BD           MOVL    PRC_L_TRMLIST(R11),R0                ;GET ADDRESS OF XABTRM ITEM
     06 68 AB    00    E1 00C1           BBC     #PRC_V_CARRCNTL,PRC_W_FLAGS(R11),2$  ;SKIP IF NO CR/LF INDICATED
  10 B0    0000'CF     B0 00C6           MOVW    DCL$CRLF,@ITRM_L_PMPTADDR(R0)        ;INSERT A CR/LF OTHERWISE
                          00CC   ;
                          00CC   ; IF IN THE MIDST OF A GOTO SCAN, THEN SKIP THE UNNECCESSARY PROCESSING.
                          00CC   ;
     09 68 AB    04    E1 00CC   2$:     BBC     #PRC_V_GOTO,PRC_W_FLAGS(R11),10$     ;SKIP IF NOT GOTO READ
               6243    94 00D1           CLRB    (R2)[R3]                             ;SET EOL
               00A8    31 00D4           BRW     RETURN                               ;RETURN
               FF6B    31 00D7   5$:     BRW     GET_INPUT                            ;
                          00DA
                          00DA   ;
                          00DA   ; CHECK FOR CTRL/B AND DEFINED KEYS.
                          00DA   ;
 16 18 A4    00000000'8F   E1 00DA   10$:    BBC     #DEV$V_TRM,RAB$L_CTX(R4),12$         ;SKIP IF NOT TERMINAL
               01DF    30 00E3           BSBW    PROCESS_RECALL                       ;CHECK FOR RECALL CHAR
          03     50    D1 00E6           CMPL    R0,#3                                ;REPROMPT REQUESTED?
                 EC    13 00E9           BEQL    5$                                   ;YES, DO IT
               0319    30 00EB           BSBW    PROCESS_ESCAPE                       ;PROCESS ESCAPE SEQUENCES
          03     50    D1 00EE           CMPL    R0,#3                                ;REPROMPT REQUESTED?
                 E4    13 00F1           BEQL    5$                                   ;YES, DO IT
               6243    94 00F3           CLRB    (R2)[R3]                             ;SET EOL INDICATOR
              B5 50    E9 00F6           BLBC    R0,ERROR1                            ;BRANCH IF ERROR
                          00F9
                          00F9   ;
                          00F9   ; IF INTERACTIVE AND IF THE RECORD IS NON-NULL, THEN COPY THE COMMAND LINE
                          00F9   ; TO THE RECALL BUFFER.
                          00F9   ;
               6243    94 00F9   12$:    CLRB    (R2)[R3]                             ;SET EOL INDICATOR
                 53    D5 00FC           TSTL    R3                                   ;NULL COMMAND?
                 31    13 00FE           BEQL    40$                                  ;YES, THEN SKIP
     2C 68 AB    06    E0 0100           BBS     #PRC_V_MODE,PRC_W_FLAGS(R11),40$     ;IF SET, BATCH JOB
     0A 68 AB    0B    E0 0105           BBS     #PRC_V_YLEVEL,PRC_W_FLAGS(R11),15$   ;IF SET, CTRL/Y LEVEL
     05 F0 AA    07    E0 010A           BBS     #WRK_V_INQUIRE,WRK_W_FLAGS(R10),15$  ;IF SET, INQUIRING
              5C AB    D5 010F           TSTL    PRC_L_INDEPTH(R11)                   ;INTERACTIVE LEVEL 0?
                 1D    12 0112           BNEQ    40$                                  ;IF NO, DON'T SAVE
                 52    DD 0114   15$:    PUSHL   R2                                   ;PUSH COMMAND ADDR
                 53    DD 0116           PUSHL   R3                                   ;PUSH COMMAND LEN
                 5E    DD 0118           PUSHL   SP                                   ;PUSH DESCR ADDR
     07 F0 AA    03    E0 011A           BBS     #WRK_V_CONTIN,WRK_W_FLAGS(R10),20$   ;IS IT CONTIN TYPE PROMPT?
      0000'CF    01    FB 011F           CALLS   #1,DCL$PUT_COMMAND                   ;SAVE THE COMMAND AWAY
                 05    11 0124           BRB     30$                                  ;BRANCH
      0000'CF    01    FB 0126   20$:    CALLS   #1,DCL$PUT_SEGMENT                   ;SAVE THE SEGMENT AWAY
            53 8ED0       012B   30$:    POPL    R3                                   ;RESTORE R3
            52 8ED0       012E           POPL    R2                                   ;RESTORE R2
                          0131
                          0131   ;
                          0131   ; SUBSTITUTE ANY SYMBOLS WHICH ARE DELIMITED BY SINGLE QUOTES
                          0131   ;
      62   53   27   3A  0131     305 40$:      LOCC    #^A/'/,R3,(R2)                          ;LINE HAVE POSSIBLE STRING S
                03   13  0135     306           BEQL    50$                                     ;IF EQL NO
              0490   30  0137     307           BSBW    EXPAND                                  ;EXPAND LINE IF APPROPRIATE
                         013A     308
                         013A     309 ;
                         013A     310 ; IF VERIFY MODE, WRITE A COPY OF THE COMMAND LINE TO THE LOG FILE
                         013A     311 ;
   1B 68 AB     07   E1  013A     312 50$:      BBC     #PRC_V_VERIFY,PRC_W_FLAGS(R11),70$      ;IF CLR, NO LINE VERIFICATIO
   16 68 AB     0B   E0  013F     313           BBS     #PRC_V_YLEVEL,PRC_W_FLAGS(R11),70$      ;IF SET, AT CONTROL Y/C LEVE
   0B 68 AB     06   E0  0144     314           BBS     #PRC_V_MODE,PRC_W_FLAGS(R11),60$        ;IF SET, BATCH JOB
           5C AB     D5  0149     315           TSTL    PRC_L_INDEPTH(R11)                      ;INTERACTIVE LEVEL 0?
                0C   13  014C     316           BEQL    70$                                     ;IF YES, DON'T ECHO
         012D CB     95  014E     317           TSTB    PRC_B_EXONLYL(R11)                      ;EXE-ONLY PROCEDURE?
                06   12  0152     318           BNEQ    70$                                     ;IF YES, DON'T ECHO
           51   53   D0  0154     319 60$:      MOVL    R3,R1                                   ;THE LENGTH OF THE LINE
             FEA6'   30  0157     320           BSBW    DCL$MSGOUT                              ;OUTPUT INPUT LINE
                         015A     321
                         015A     322 ;
                         015A     323 ; IF WE JUST READ A FULL-LINE COMMENT, RE-ISSUE READ NOW AS AN OPTIMIZATION.
                         015A     324 ;
   20 68 AB     01   E0  015A     325 70$:      BBS     #PRC_V_CNTRLY,PRC_W_FLAGS(R11),RETURN   ;MUST TAKE IMMEDIATE ACTION
                         015F     326                                                           ;IF CTRLY HIT - SKIP OPTIMIZ
                54   D4  015F     327           CLRL    R4                                      ;SET NO DOLLAR SIGN SEEN FLA
 96'AF     05   82   3A  0161     328 80$:      LOCC    (R2)+,#5,B^SPECIAL                      ;CHECK FOR SPECIAL CHARS
                17   13  0166     329           BEQL    RETURN                                  ;IF EQL NO MATCH
           50   03   C2  0168     330           SUBL    #3,R0                                   ;BLANK OR TAB?
                F4   19  016B     331           BLSS    80$                                     ;IF LSS YES
                10   13  016D     332           BEQL    RETURN                                  ;IF EQL END OF LINE
                50   D7  016F     333           DECL    R0                                      ;DOLLAR SIGN OR COMMENT?
                08   1B  0171     334           BLEQU   90$                                     ;IF LEQU DOLLAR SIGN
   07 68 AB     05   E0  0173     335           BBS     #PRC_V_IND,PRC_W_FLAGS(R11),RETURN      ;BRANCH IF FLUSHING RECORD
              FE8C   31  0178     336           BRW     REINP                                   ;GET NEXT RECORD
      E2 54     00   E3  017B     337 90$:      BBCS    #0,R4,80$                               ;IF CLR, FIRST DOLLAR SIGN
                         017F     338
                         017F     339 ;
                         017F     340 ; IF THE PREVIOUS RECORD ENDED WITH TRAILING SPACES OR TABS,
                         017F     341 ; INSERT A SPACE AT THE FRONT OF THE CURRENT INPUT RECORD SO
                         017F     342 ; THAT PARAMETERS ARE DELIMITED PROPERLY.
                         017F     343 ;
   06 F0 AA     09   E5  017F     344 RETURN:   BBCC    #WRK_V_TRAILSPC,WRK_W_FLAGS(R10),10$    ;IF CLR, NO TRAILING SPACE S
           50   20   90  0184     345           MOVB    #^A/ /,R0                               ;SET SPACE CHARACTER
             FE76'   30  0187     346           BSBW    DCL$BACKUPCHAR                          ;APPEND TO FRONT OF INPUT BU
                1C   BA  018A     347 10$:      POPR    #^M<R2,R3,R4>                           ;RESTORE REGISTERS
 00F3 CB   5F   8F   90  018C     348           MOVB    #^A/ /,PRC_B_CONTINUE(R11)              ;SET FOR CONTINUATION PROMPT
             FE6B'   30  0192     349           BSBW    DCL$GETCHAR                             ;GET FIRST CHARACTER OF NEW
                     05  0195     350           RSB
                         0196     351
                         0196     352 SPECIAL:
     09 20 00 24 21      0196     353           .ASCII  /!$/<0>/        /                       ;SPECIAL CHARACTERS
```

```
                           019B  ;
                           019B  ; AN INPUT I/O ERROR HAS OCCURRED.  IF WE GOT "RECORD STREAM ACTIVE",
                           019B  ; TRY REPEATING THE READ 1000 TIMES.  IF THAT FAILS, TRY CANCELING ALL
                           019B  ; I/O ON THE INPUT CHANNEL AND THEN REISSUING THE READ.
                           019B  ;
                           019B  IO_ERROR:
 0000'8F     50      B1    019B          CMPW    R0,#RMS$_RSA&^XFFFF                     ;ERROR RECORD STREAM ACTIVE
             48      12    01A0          BNEQ    30$                                     ;IF NO CHECK FOR END_OF_FILE
                           01A2
                           01A2          ;
                           01A2          ; WE DON'T WANT TO HAVE TO CANCEL AN RMS I/O UNLESS NECESSARY.
                           01A2          ; RETRY UP SEVERAL TIMES TO ALLOW CURRENT WRITE I/O TO COMPLETE.
                           01A2          ; THIS AVOIDS SPURIOUS WRITE ABORT MESSAGES FROM USER PROGRAMS,
                           01A2          ; CAUSED BY CANCELING AN RMS I/O OPERATION.
                           01A2          ;
  52    03E8 8F      3C    01A2          MOVZWL  #1000,R2                                ;SETUP RETRY COUNT
 05 F0 AA    07      E1    01A7          BBC     #WRK_V_INQUIRE,WRK_W_FLAGS(R10),10$     ;SKIP IF NOT INQUIRING
                           01AC          SETBIT  RAB$V_PPF_IND,RAB$W_ISI(R4)             ;SET INDIRECT PPF
                           01B1  10$:    $GET    RAB=(R4)                                ;SEE IF THE WAITING GAVE RMS
                           01BA          CLRBIT  RAB$V_PPF_IND,RAB$W_ISI(R4)             ;CLEAR INDIRECT PPF
          20 50      E8    01BF          BLBS    R0,20$                                  ;THE WAIT DID THE TRICK, GO
 0000'8F     50      B1    01C2          CMPW    R0,#RMS$_RSA&^XFFFF                     ;RECORD STREAM STILL ACTIVE?
             21      12    01C7          BNEQ    30$                                     ;BRANCH IF NOT
          E5 52      F5    01C9          SOBGTR  R2,10$                                  ;RETRY UNTIL STREAM BECOMES
                           01CC
                           01CC          ;
                           01CC          ; CONSTANT RETRYING DIDN'T HELP.  CANCEL ANY I/O AND TRY AGAIN.
                           01CC          ;
                           01CC          $CANCEL_S PRC_W_INPCHAN(R11)                    ;IF NO LUCK, STOP THE I/O ON
                           01D7          $WAIT    RAB=(R4)                               ;WAIT FOR I/O TO COMPLETE
             37      11    01E0          BRB     40$                                     ;TRY TO READ AGAIN
                           01E2
                           01E2  ;
                           01E2  ; READ FINALLY SUCCEEDED.  PROCESS THE INPUT RECORD.
                           01E2  ;
  52    F894 CA      9E    01E2  20$:    MOVAB   WRK_G_INPBUF-2(R10),R2                  ;GET ADDRESS OF INPUT BUFFER
           FEC7      31    01E7          BRW     PROCESS_INPUT                           ;PROCESS THE INPUT LINE
                           01EA
                           01EA  ;
                           01EA  ; WE HAVE ENCOUNTERED AN ERROR OTHER THAN RECORD STREAM ACTIVE.  RESTORE THE
                           01EA  ; DEFAULT READ FORMAT OF NO INITIAL STRING, NO OFFSET AND RESTORE THE DEFAULT
                           01EA  ; KEYPAD STATE.
                           01EA  ;
    51   40 A4       D0    01EA  30$:    MOVL    RAB$L_XAB(R4),R1                        ;GET ADDRESS OF XABTRM
    0C A1    18      D0    01EE          MOVL    #ITRM_K_MINLEN,XAB$W_ITMLST_LEN(R1)     ;SET SHORT LENGTH OF ITEM LI
           03BC      30    01F2          BSBW    DCL$LOCKED_STATE                        ;RESTORE LOCKED KEYPAD STATE
                           01F5
                           01F5  ;
                           01F5  ; IF WE ARE AT END OF FILE, TERMINATE THE CURRENT PROCEDURE LEVEL AND
                           01F5  ; READ THE NEXT RECORD FROM THE PREVIOUS PROCEDURE LEVEL.
                           01F5  ;
 0000'8F     50      B1    01F5          CMPW    R0,#RMS$_EOF&^XFFFF                     ;END OF FILE?
             58      12    01FA          BNEQ    110$                                    ;IF NEQ NO
             62      94    01FC          CLRB    (R2)                                    ;SET END OF LINE INDICATOR
 28 68 AB    0B      E0    01FE          BBS     #PRC_V_YLEVEL,PRC_W_FLAGS(R11),50$      ;IF SET, AT CONTROL Y/C LEVE
 23 F0 AA    07      E0    0203          BBS     #WRK_V_INQUIRE,WRK_W_FLAGS(R10),50$     ;BR IF DOING AN INQUIRE
          5C AB      D5    0208          TSTL    PRC_B_INDEPTH(R11)                      ;INDIRECT LEVEL ZERO?
```

```
                   21   13  020B   412              BEQL    70$                                     ;IF EQL, YES
                FDF0'   30  020D   413              BSBW    DCL$UNSTACK                             ;UNSTACK INDIRECT FILE
                            0210   414
                            0210   415 ;
                            0210   416 ; IF WE JUST RETURNED BACK TO LEVEL 0, FORCE THE PROMPT STRING BACK TO NORMAL
                            0210   417 ; BEFORE RE-ISSUING A READ FOR THE NEXT COMMAND.
                            0210   418 ;
                 5C AB  D5  0210   419              TSTL    PRC_L_INDEPTH(R11)                      ;INDIRECT LEVEL ZERO?
                    04  12  0213   420              BNEQ    40$                                     ;BRANCH IF NOT
              00F3 CB   94  0215   421              CLRB    PRC_B_CONTINUE(R11)                     ;SET FOR NORMAL PROMPT
                            0219   422
                            0219   423 ;
                            0219   424 ; IF CURRENTLY PROCESSING A LINE CONTINUATION, RETURN AN EOL CHARACTER.
                            0219   425 ;
                            0219   426 ; IF INDIRECT FILE RECOGNITION IS DISABLED, THIS IS A FLUSH OF A COMMAND WITH
                            0219   427 ; "-" AS THE LAST CHARACTER.  RETURN AN EOL CHARACTER.  OTHERWISE, PROCESS
                            0219   428 ; THE LINE JUST READ
                            0219   429 ;
       0D 68 AB    06   E0  0219   430 40$:         BBS     #PRC_V_FLUSH,PRC_W_FLAGS(R11),50$       ;IF BIT IS SET,THEN FLUSH
       08 68 AB    05   E0  021E   431              BBS     #PRC_V_IND,PRC_W_FLAGS(R11),50$         ;IF BIT IS SET,THEN FLUSH
       03 F0 AA    03   E0  0223   432              BBS     #WRK_V_CONTIN,WRK_W_FLAGS(R10),50$      ;IF BIT IS SET,THEN FLUSH
                 FDDC   31  0228   433              BRW     REINP                                   ;READ THE NEXT LINE
                 FF51   31  022B   434 50$:         BRW     RETURN                                  ;FLUSH THE RECORD
                            022E   435
                            022E   436 ;
                            022E   437 ; IF WE GOT AN END OF FILE WHILE READING THE LEVEL 0 PROCEDURE IN A
                            022E   438 ; NON-INTERACTIVE JOB, TERMINATE THE JOB STEP.  IGNORE EOF'S (CTRL/Z)
                            022E   439 ; IN AN INTERACTIVE JOB.
                            022E   440 ;
       14 68 AB    04   E5  022E   441 70$:         BBCC    #PRC_V_GOTO,PRC_W_FLAGS(R11),90$        ;BR IF NOT IN A GOTO
                FDCA'   30  0233   442              BSBW    DCL$DEALGOTO                            ;DEALLOCATE GOTO SYMBOL
                            0236   443              STATUS  USGOTO                                  ;SET FINAL STATUS OF UNSTATI
                            023D   444              SETBIT  WRK_V_COMMAND,WRK_W_FLAGS(R10)          ;MARK COMMAND EXECUTION ERRO
                            0241   445              ERRMSG                                          ;PRINT THE ERROR
                FDB9'   30  0244   446 80$:         BSBW    DCL$SET_STATUS                          ;GIVE ERROR HANDLER A CHANCE
          68 AB    0E   E0  0247   447 90$:         BBS     #PRC_V_EOFLOGO,PRC_W_FLAGS(R11),-       ;IF SILENT LOGOUT REQUESTED
                   4F       024B   448                      SILENT_LOGOUT                           ;
       C8 68 AB    06   E1  024C   449              BBC     #PRC_V_MODE,PRC_W_FLAGS(R11),40$        ;IF NOT BATCH JOB, IGNORE EO
                FDAC'   31  0251   450              BRW     DCL$ABORT                               ;LOG OUT BATCH JOB
                            0254   451
                            0254   452 ;
                            0254   453 ; SOME OTHER TYPE OF I/O ERROR HAS OCCURRED.  ISSUE AN ERROR MESSAGE,
                            0254   454 ; THEN TERMINATE THE CURRENT PROCEDURE LEVEL AND PARSE THE NEXT COMMAND.
                            0254   455 ; IF THE ERROR WAS INSUFFICIENT QUOTA, THEN CANCEL THE CTRL/Y AST, OUTPUT
                            0254   456 ; THE ERROR MESSAGE, AND THEN LOG THE PROCESS OUT.
                            0254   457 ;
                            0254   458 110$:        SETBIT  WRK_V_COMMAND,WRK_W_FLAGS(R10)          ;MARK COMMAND EXECUTION ERRO
   00000000'8F     50   D1  0258   459              CMPL    R0,#SS$_EXQUOTA                         ;IS ERROR DUE TO EXCEEDED QU
                   2C   13  025F   460              BEQL    ABORT                                   ;YES, THEN BRANCH
                            0261   461              ASSUME  PRC_L_STV EQ PRC_L_STS+4
                            0261   462              ASSUME  RAB$L_STV EQ RAB$L_STS+4
       0084 CB   08 A4  7D  0261   463              MOVQ    RAB$L_STS(R4),PRC_L_STS(R11)            ;STORE STS AND STV VALUES
                            0267   464 120$:        ERRMSG                                          ;OUTPUT ERROR MESSAGE
       18 68 AB    0B   E0  026A   465              BBS     #PRC_V_YLEVEL,PRC_W_FLAGS(R11),STATUS   ;IF SET, CNTL Y/C LEVEL
       13 F0 AA    07   E0  026F   466              BBS     #WRK_V_INQUIRE,WRK_W_FLAGS(R10),STATUS  ;SKIP IF IN INQUIRE
                5C AB   D5  0274   467              TSTL    PRC_L_INDEPTH(R11)                      ;INDIRECT LEVEL ZERO?
                   07   12  0277   468              BNEQ    130$                                    ;IF NEQ NO
                C6 68 AB   06   E0 0279             BBS     #PRC_V_MODE,PRC_W_FLAGS(R11),80$        ;BR IF BATCH
                           07   11 027E             BRB     STATUS                                  ;
                           50   DD 0280   130$:     PUSHL   R0                                      ;SAVE ERROR/STATUS VALUE
                        FD7B'   30 0282             BSBW    DCL$UNSTACK                             ;UNSTACK INDIRECT FILE
                           01   BA 0285             POPR    #^M<R0>                                 ;RESTORE ERROR/STATUS VALUE
                        FD76'   30 0287   STATUS:   BSBW    DCL$SET_STATUS                          ;SET COMPLETION STATUS
                        FD73'   31 028A             BRW     DCL$RESTART                             ;
                                   028D
```

```
                     028D  ;
                     028D  ; OUTPUT AN ERROR MESSAGE AND LOG THE PROCESS OUT
                     028D  ;
         50   DD     028D  ABORT:  PUSHL   R0                                          ;SAVE THE STATUS
      FD6E'   30     028F          BSBW    DCL$DSBCONTRLY                              ;CANCEL THE CTRL/Y AST
         50 8ED0     0292          POPL    R0                                          ;RESTORT THE STATUS
                     0295          ERRMSG                                              ;OUTPUT THE ERROR MESSAGE
      FD65'   31     0298          BRW     DCL$ABORT                                   ;LOG THE PROCESS OUT
                     029B
                     029B  ;
                     029B  ; PERFORM A SILENT LOGOUT BY CANCELING THE SUPERVISOR MODE EXIT HANDLERS
                     029B  ; (SO THAT THE PROCESS IS DELETED), AND INVOKING $EXIT WITH THE LATEST
                     029B  ; STATUS FOR THIS PROCESS.
                     029B  ;
                     029B  SILENT_LOGOUT::
      FD62'   30     029B          BSBW    DCL$CLOSE_PPFS                              ;CLOSE ALL PPF FILES STILL O
 50    1C AB   D0    029E          MOVL    PRC_L_INDFAB(R11),R0                        ;GET ADDR OF INDIRECT FAB
    0114 CB    B0    02A2          MOVW    PRC_W_OUTIFI(R11),-                         ;GET INTERNAL FILE INDEX OF
      02 A0          02A6                  FAB$W_IFI(R0)                               ;
                     02A8          $CLOSE  FAB=(R0)                                    ;CLOSE INDIRECT OUTPUT FILE
                     02B1          $CANEXH_S                                           ;CANCEL SUPERVISOR MODE EXIT
                     02BA          $EXIT_S PRC_L_LSTSTATUS(R11)                        ;EXIT PROCESS WITH FINAL STA
```

```
                         02C5          .SBTTL  PROCESS RECALL COMMANDS
                         02C5  ;---
                         02C5  ; PROCESS_RECALL - PROCESS RECALL COMMANDS
                         02C5  ;
                         02C5  ; SUBROUTINE TO CHECK FOR AND PROCESS CTRL/B AND ARROW RECALL COMMANDS.
                         02C5  ;
                         02C5  ; INPUTS:
                         02C5  ;
                         02C5  ;       R2 = ADDRESS OF NEW INPUT RECORD
                         02C5  ;       R3 = LENGTH OF NEW INPUT RECORD, EXCLUDING NULL AT END OF LINE
                         02C5  ;       R4 = ADDRESS OF RAB
                         02C5  ;
                         02C5  ; OUTPUTS:
                         02C5  ;
                         02C5  ;       INPUT BUFFER IS INITIALIZED
                         02C5  ;       R0 IS SET
                         02C5  ;---
                         02C5  PROCESS_RECALL:                                 ;PROCESS RECALL COMMAND
                         02C5
                         02C5  ;
                         02C5  ; CHECK FOR CTRL/B.
                         02C5  ;
    0C A4    02     91   02C5          CMPB    #^X02,RAB$W_STV0(R4)            ;TERMINATOR CTRL/B?
             04     13   02C9          BEQL    RECALL_PREV                     ;YES, THEN PROCESS IT
       50    01     D0   02CB          MOVL    #1,R0                           ;RETURN
                    05   02CE          RSB                                     ;
                         02CF
                         02CF  ;
                         02CF  ; REPROMPT WITH THE PREVIOUS COMMAND.
                         02CF  ;
                         02CF  RECALL_PREV:
     02DF           30   02CF          BSBW    DCL$LOCKED_STATE                ;RESTORE LOCKED STATE
       C5    AA     95   02D2          TSTB    WRK_B_RECALLCNT(R10)            ;FIRST TIME?
             16     13   02D5          BEQL    10$                             ;YES, THEN ALWAYS GO AHEAD
       EA    BA     95   02D7          TSTB    @WRK_L_RECALLPTR(R10)           ;POINTER ADJUST. BY RECALL_NEXT?
             03     13   02DA          BEQL    5$                              ;NO, OK AS IS
       EA    AA     D7   02DC          DECL    WRK_L_RECALLPTR(R10)            ;YES, SET IT BACK TO NORMAL
                         02DF  ;
     012F    CB     D1   02DF  5$:     CMPL    PRC_L_RECALLPTR(R11),-          ;RETURNED TO THE ORIGIN?
       EA    AA          02E3                  WRK_L_RECALLPTR(R10)            ;
             3B     13   02E5          BEQL    END_OF_LIST                     ;
       C5    AA     91   02E7          CMPB    WRK_B_RECALLCNT(R10),-          ;MAX # OF COMMANDS DISPLAYED?
             15          02EA                  #WRK_C_RECALLMAX+1              ;
             35     13   02EB          BEQL    END_OF_LIST                     ;
             7E     7C   02ED  10$:    CLRQ    -(SP)                           ;ALLOCATE A DESCRIPTOR
             5E     DD   02EF          PUSHL   SP                              ;PUSH DESCR ADDRESS
  0000'CF    01     FB   02F1          CALLS   #1,DCL$GET_PREV_COMMAND         ;RECALL THE SPECIFIED COMMAND
       17    50     E9   02F6          BLBC    R0,20$                          ;BRANCH IF NO COMMAND WAS FOUND
       C5    AA     96   02F9          INCB    WRK_B_RECALLCNT(R10)            ;INCR RECALL COUNT
  50 0133    CB     9E   02FC          MOVAB   PRC_G_COMMANDS(R11),R0          ;R0 = ADDR. OF BEGINNING OF BUFFER
    EA AA    50     D1   0301          CMPL    R0,@RK_L_RECALLPTR(R10)         ;ARE WE AT BEGINNING OF BUFFER?
             03     12   0305          BNEQ    15$
       EA    AA     D6   0307          INCL    WRK_L_RECALLPTR(R10)            ;YES, FIX POINTER TO INDICATE
                         030A                                                  ;THIS TO RECALL NEXT.
       C5    AA     91   030A  15$:    CMPB    WRK_B_RECALLCNT(R10),-          ;MAX # OF COMMANDS DISPLAYED?
             15          030D                  #WRK_C_RECALLMAX+1              ;
             05     12   030E          BNEQ    RECALL_CURR                     ;BRANCH IF NOT NOW MAX
```

```
          5E   08   C0  0310  20$:    ADDL    #8,SP                           ;RESTORE THE STACK
               0D   11  0313          BRB     END_OF_LIST                     ;OUTPUT A BLANK LINE
                        0315
                        0315  ;
                        0315  ; REPROMPT WITH THE COMMAND THAT IS ON THE STACK.
                        0315  ;
                        0315  RECALL_CURR:
             0067   30  0315          BSBW    ERASE_LINE                      ;ERASE THE PREVIOUS LINE
          51   8E   7D  0318          MOVQ    (SP)+,R1                        ;GET COMMAND DESCRIPTOR
             00D1   30  031B          BSBW    INSERT_COMMAND                  ;MODIFY THE XABTRM
          50   03   D0  031E          MOVL    #3,R0                           ;SET REPROMPT RETURN STATUS
                    05  0321          RSB                                     ;RETURN
                        0322
                        0322  ;
                        0322  ; REPROMPT WITH A BLANK LINE.
                        0322  ;
                        0322  END_OF_LIST:
             005A   30  0322          BSBW    ERASE_LINE                      ;ERASE THE PREVIOUS LINE
               51   7C  0325          CLRQ    R1                              ;INIT COMMAND DESCRIPTOR
             00C5   30  0327          BSBW    INSERT_COMMAND                  ;REPROMPT WITH THIS COMMAND
          50   03   D0  032A          MOVL    #3,R0                           ;SET REPROMPT RETURN STATUS
                    05  032D          RSB                                     ;RETURN
                        032E
                        032E  ;
                        032E  ; REPROMPT WITH THE NEXT COMMAND.
                        032E  ;
                        032E  ; NOTE: WHEN AT BEGINNING OF BUFFER (NULL PRECEDING CURRENT RECORD),
                        032E  ;       SPECIAL PROCESSING IS DOWN. THE DCL$GET CURR COMMAND IS USED
                        032E  ;       INSTEAD OF DCL$GET NEXT COMMAND. THE WRK RECALL POINTER IS USED
                        032E  ;       AS A FLAG TO DETERMINE WHETHER OR NOT THE FIRST COMMAND IN THE
                        032E  ;       BUFFER HAS BEEN ALREADY PROCESSED. NORMALLY, THE POINTER IS POINTING
                        032E  ;       TO THE BEGINNING OF A COMMAND (FIRST BYTE ALWAYS NULL). IF THE
                        032E  ;       POINTER IS POINTING TO A NON-NULL BYTE, IT MEANS THAT THE FIRST
                        032E  ;       COMMAND IN THE BUFFER HAS ALREADY BEEN PROCESSED(THE POINTER WILL
                        032E  ;       BE POINTING TO THE LENGTH BYTE FOLLOWING THE NULL ).
                        032E  ;       IN THIS CASE, THE POINTER IS CORRECTED AND THE COMMAND IS
                        032E  ;       RETRIEVED WITH THE DCL$GET_NEXT_COMMAND INSTEAD OF DCL$GET_CURR_COMMAND.
                        032E  ;
                        032E  RECALL_NEXT:
             0280   30  032E          BSBW    DCL$LOCKED_STATE                ;RESTORE LOCKED STATE
            C5 AA   95  0331          TSTB    WRK_B_RECALLCNT(R10)            ;ANY NEXT COMMANDS TO RECALL?
               EC   13  0334          BEQL    END_OF_LIST                     ;NO, THEN REPEAT CURRENT COMMAND
                        0336  ;
            EA BA   95  0336          TSTB    @WRK_L_RECALLPTR(R10)           ;POINTING TO BEGINNING OF COMMAND?
               05   13  0339          BEQL    25$                             ;YES, HANDLE NORMALLY
                        033B  ;
            EA AA   D7  033B          DECL    WRK_L_RECALLPTR(R10)            ;NO, ADJUST POINTER
               2C   11  033E          BRB     30$                             ;AND GET NEXT COMMAND.
                        0340  ;
   50    EA AA   01   C3  0340  25$:    SUBL3   #1,WRK_L_RECALLPTR(R10),R0      ;POINT R0 TO CURR. COMMAND-1
      51    0133   CB   9E  0345          MOVAB   PRC_G_COMMANDS(R11),R1          ;R1 = ADDR. OF BEGINNING OF BUF.
          51   50   D1  034A          CMPL    R0,R1                           ;BUFFER UNDERFLOW?
               05   1E  034D          BGEQU   28$                             ;NO, R0 OK AS IS
      50    0401   C0   9E  034F          MOVAB   PRC_C_CMDBUFSIZ(R0),R0          ;YES, POINT R0 TO TOP OF BUFFER
               60   95  0354  28$:    TSTB    (R0)                            ;PREVIOUS COMMAND EXIST?
               14   12  0356          BNEQ    30$                             ;YES, PROCESS NORMALLY
                        0358  ;
                        0358   615 ;       SPECIAL CASE: THERE ARE EXACTLY 21 COMMANDS IN THE COMMAND BUFFER
                        0358   616 ;       AND RECALL_PREV HAS RETRIEVED THE LAST VALID COMMAND (IT LEAVES
                        0358   617 ;       POINTER TO FIRST COMMAND.) IN THIS CASE, DO A GET NEXT INSTEAD OF
                        0358   618 ;       GET CURRENT.
                        0358   619 ;
          C5 AA     91  0358   620         CMPB    WRK_B_RECALLCNT(R10),-          ;HAVE MAX. COMMANDS BEEN DISPL?
             15         035B   621                 #WRK_C_RECALLMAX+1
             0E     13  035C   622         BEQL    30$                             ;YES, GET NEXT COMMAND.
                        035E   623 ;
             7E     7C  035E   624         CLRQ    -(SP)                           ;NO, ALLOCATE A DESCRIPTOR
             5E     DD  0360   625         PUSHL   SP                              ;PUSH DESCRIPTOR ADDR.
   0000'CF   01     FB  0362   626         CALLS   #1,DCL$GET_CURR_COMMAND         ;GET CURRENT COMMAND
          EA AA     D6  0367   627         INCL    WRK_L_RECALLPTR(R10)            ;INDICATE 1ST COMMAND PROCESSED.
             A9     11  036A   628         BRB     RECALL_CURR                     ;OUTPUT CURRENT COMMAND
                        036C   629 ;
             7E     7C  036C   630 30$:    CLRQ    -(SP)                           ;ALLOCATE A DESCRIPTOR
             5E     DD  036E   631         PUSHL   SP                              ;PUSH DESCR ADDRESS
   0000'CF   01     FB  0370   632         CALLS   #1,DCL$GET_NEXT_COMMAND         ;RECALL THE SPECIFIED COMMAND
          C5 AA     97  0375   633         DECB    WRK_B_RECALLCNT(R10)            ;DECREMENT THE RECALL COUNT
             9B     12  0378   634         BNEQ    RECALL_CURR                     ;BRANCH IF NOT NOW ZERO
       5E    08     C0  037A   635         ADDL    #8,SP                           ;RESTORE THE STACK
             A3     11  037D   636         BRB     END_OF_LIST                     ;OUTPUT A BLANK LINE
```

```
                           037F  ;
                           037F  ; ERASE THE CURRENT LINE IF THE TERMINAL IS ANSI CRT.
                           037F  ;
                           037F  ;        R0,R5 ARE DESTROYED
                           037F  ;
                           037F  ERASE_LINE:
            7E     7C      037F          CLRQ     -(SP)                          ;CREATE ITEM LIST
         F8 AE     9F      0381          PUSHAB   -8(SP)                         ;SET BUFFER ADDRESS
   001C0004 8F     DD      0384          PUSHL    #DVI$_DEVDEPEND2@16+4          ;SET ITEM CODE
            7E     7C      038A          CLRQ     -(SP)                          ;ALLOCATE AN IOSB
      50    5E     D0      038C          MOVL     SP,R0                          ;GET ADDRESS OF IOSB
                           038F          $GETDVIW_S  EFN=#EXE$C_SYSEFN,-         ;GET DEVDEPEND2
                           038F                   CHAN=PRC_W_INPCHAN(R11),-      ;
                           038F                   ITMLST=8(R0),-                 ;
                           038F                   IOSB=(R0)                      ;
         3D 50     E9      03AB          BLBC     R0,90$                         ;BRANCH IF ERROR
      50    6E     3C      03AE          MOVZWL   (SP),R0                        ;GET IOSB STATUS
         37 50     E9      03B1          BLBC     R0,90$                         ;BRANCH IF ERROR
                           03B4
   32 08 AE    18  E1      03B4          BBC      #TT2$V_ANSICRT,8(SP),90$       ;BRANCH IF ANSI CRT BIT CLEAR
      55    0C AB  D0      03B9          MOVL     PRC_L_OUTRAB(R11),R5           ;SET ADDRESS OF OUTPUT FILE RAB
  28 A5   FC40 CF  9E      03BD  10$:    MOVAB    ERASE+1,RAB$L_RBF(R5)          ;SET ADDRESS OF OUTPUT RECORD
  22 A5   FC39 CF  9B      03C3          MOVZBW   ERASE,RAB$W_RSZ(R5)            ;SET SIZE OF OUTPUT RECORD
                           03C9          DISABLE                                 ;DISABLE CONTROL Y/C AST'S
      06      00   F0      03CD          INSV     #0,#RAB$V_PPF_RAT,-            ;DISABLE CR FORMAT WRITES
   02 A5      08           03D0                   #RAB$S_PPF_RAT,RAB$W_ISI(R5)   ;
                           03D3          $PUT     RAB=(R5)                       ;OUTPUT RECORD
      06      02   F0      03DC          INSV     #FAB$M_CR,#RAB$V_PPF_RAT,-     ;ENABLE CR FORMAT WRITES
   02 A5      08           03DF                   #RAB$S_PPF_RAT,RAB$W_ISI(R5)   ;
                           03E2          ENABLE                                  ;ENABLE CONTROL Y/C AST'S
                           03E4
      50    10 AB  D0      03E4          MOVL     PRC_L_TRMLIST(R11),R0          ;GET ADDRESS OF XABTRM ITEM LIST
            10 B0  B4      03E8          CLRW     @ITRM_L_PMPTADDR(R0)           ;REMOVE CR/LF FROM PROMPT STRING
                           03EB
      5E    18     C0      03EB  90$:    ADDL     #6*4,SP                        ;RESTORE STACK
                   05      03EE          RSB                                     ;
```

```
                         03EF   675 ;
                         03EF   676 ; REPROMPT WITH RECALLED COMMAND.
                         03EF   677 ;
                         03EF   678 INSERT_COMMAND:
   50    10 AB     D0    03EF   679         MOVL    PRC_L_TRMLIST(R11),R0        ;GET ADDRESS OF XABTRM ITEM LIST
         28 A0     D4    03F3   680         CLRL    ITRM_L_OFFSET(R0)            ;REQUEST A FRESH READ
   18 A0    51     B0    03F6   681         MOVW    R1,ITRM_W_INILEN(R0)         ;SET LENGTH OF INITIAL STRING
   1C A0    52     D0    03FA   682         MOVL    R2,ITRM_L_INIADDR(R0)        ;SET ADDRESS OF INITIAL STRING
   50    40 A4     D0    03FE   683         MOVL    RAB$L_XAB(R4),R0             ;GET ADDRESS OF XABTRM
            30     D0    0402   684         MOVL    #ITRM_K_LENGTH,-             ;SET EXPANDED LENGTH OF ITEM LIST
         0C A0           0404   685                 XAB$W_ITMLST_LEN(R0)         ;
                   05    0406   686         RSB                                  ;RETURN
```

```
                           0407        .SBTTL  PROCESS ESCAPE SEQUENCES
                           0407  ;---
                           0407  ; PROCESS_ESCAPE - PROCESS ESCAPE SEQUENCES
                           0407  ;
                           0407  ; SUBROUTINE TO CHECK FOR AND PROCESS ESCAPE SEQUENCES.
                           0407  ;
                           0407  ; INPUTS:
                           0407  ;
                           0407  ;       R2 = ADDRESS OF NEW INPUT RECORD
                           0407  ;       R3 = LENGTH OF NEW INPUT RECORD, EXCLUDING NULL AT END OF LINE
                           0407  ;       R4 = ADDRESS OF RAB
                           0407  ;
                           0407  ; OUTPUTS:
                           0407  ;
                           0407  ;       R2 = ADDRESS OF NEW INPUT RECORD
                           0407  ;       R3 = LENGTH OF NEW INPUT RECORD, EXCLUDING NULL AT END OF LINE
                           0407  ;---
                           0407
                           0407  PROCESS_ESCAPE:                                         ;PROCESS ESCAPE SEQUENCES
                           0407
                           0407  ;
                           0407  ; CHECK FOR ESCAPE SEQUENCES.
                           0407  ;
      0E A4    01    91    0407          CMPB    #1,RAB$W_STV2(R4)               ;ESCAPE SEQUENCE?
               19    1F    040B          BLSSU   5$                              ;YES, THEN PROCESS
             01A1    30    040D          BSBW    DCL$LOCKED_STATE                ;RESTORE LOCKED KEYPAD STATE
         50    01    D0    0410          MOVL    #1,R0                           ;SET REPROMPT STATUS
                     05    0413          RSB                                     ;RETURN
                           0414
                           0414  ;
                           0414  ; REPROMPT IF NO KEY OR SYMBOL FOUND.
                           0414  ;
         54    8E    7D    0414  70$:    MOVQ    (SP)+,R4                        ;RESTORE R4/R5
         52    8E    7D    0417  80$:    MOVQ    (SP)+,R2                        ;RESTORE R2/R3
         51    53    D0    041A          MOVL    R3,R1                           ;CREATE R1/R2 DESCRIPTOR
               D0    10    041D          BSBB    INSERT_COMMAND                  ;REPROMPT WITH THIS COMMAND
             018F    30    041F          BSBW    DCL$LOCKED_STATE                ;RESTORE LOCKED KEYPAD STATE
         50    03    D0    0422          MOVL    #3,R0                           ;SET REPROMPT STATUS
                     05    0425          RSB                                     ;RETURN
                           0426
                           0426  ;
                           0426  ; FIND ASSOCIATED META-KEY NAME.
                           0426  ;
         7E    52    7D    0426  5$:     MOVQ    R2,-(SP)                        ;SAVE R2/R3
               7E    D4    0429          CLRL    -(SP)                           ;ALLOCATE SPACE FOR RETURN ADDRESS
               5E    DD    042B          PUSHL   SP                              ;POINT AT IT
      7E    0E A4    9A    042D          MOVZBL  RAB$W_STV2(R4),-(SP)            ;GET TERMINATOR LENGTH
             6243    9F    0431          PUSHAB  (R2)[R3]                        ;GET TERMINATOR ADDRESS
00000000'GF    03    FB    0434          CALLS   #3,G^GET_KEY_NAME               ;LOOK UP THE ESCAPE SEQUENCE
         52    8E    D0    043B          MOVL    (SP)+,R2                        ;GET ADDRESS OF ASCIC META-KEY NAME
         D6 50       E9    043E          BLBC    R0,80$                          ;SKIP IF NONE
                           0441
                           0441  ;
                           0441  ; SPECIAL CASE UP AND DOWN ARROW KEYS.
                           0441  ;
         04    62    91    0441          CMPB    (R2),#4                         ;IS THE STRING SHORT ENOUGH?
               1E    14    0444          BGTR    10$                             ;NO, THEN SKIP
```

```
     5055 8F    01 A2    B1  0446          CMPW    1(R2),#^A'UP'                  ;CHECK FOR UP ARROW
                   10    13  044C          BEQL    8$                             ;BRANCH IF MATCH
 4E574F44 8F    01 A2    D1  044E          CMPL    1(R2),#^A'DOWN'                ;CHECK FOR DOWN ARROW
                   0C    12  0456          BNEQ    10$                            ;BRANCH IF NO MATCH
           52      8E    7D  0458  7$:     MOVQ    (SP)+,R2                       ;RESTORE R2/R3
                 FED0    31  045B          BRW     RECALL_NEXT                    ;DO THE RECALL
           52      8E    7D  045E  8$:     MOVQ    (SP)+,R2                       ;RESTORE R2/R3
                 FE6B    31  0461          BRW     RECALL_PREV                    ;DO THE RECALL
                             0464
                             0464  ;
                             0464  ; FIND ASSOCIATED SYMBOL VALUE.
                             0464  ;
                             0464  ;       (SP) = INITIAL OFFSET
                             0464  ;       4(SP) = NEW COMMAND LENGTH
                             0464  ;       8(SP) = RAB ADDRESS
                             0464  ;       12(SP) = R5
                             0464  ;       16(SP) = INITIAL COMMAND ADDRESS
                             0464  ;       20(SP) = INITIAL (LATER NEW) COMMAND LENGTH
                             0464  ;
           7E      54    7D  0464  10$:    MOVQ    R4,-(SP)                       ;SAVE R4/R5
           51      82    9A  0467          MOVZBL  (R2)+,R1                       ;GET DESCRIPTOR OF SYMBOL NAME
                FB93'    30  046A          BSBW    DCL$SEARCH_KEYPAD              ;SEARCH SYMBOL TABLE FOR MATCH
              A4 50      E9  046D          BLBC    R0,70$                         ;SKIP IF NOT FOUND
                             0470
                             0470  ;
                             0470  ; /ERASE - COPY STRING INTO INPUT BUFFER.
                             0470  ;
        2A 54      04    E1  0470          BBC     #SYM_V_ERASE,R4,20$            ;IF /NOERASE, THEN SKIP
                   51    DD  0474          PUSHL   R1                             ;SAVE NEW LENGTH
                   7E    D4  0476          CLRL    -(SP)                          ;SET INITIAL OFFSET
           55   10 AE    D0  0478          MOVL    16(SP),R5                      ;GET INITIAL ADDRESS
                   54    DD  047C          PUSHL   R4                             ;SAVE FLAGS
       65    62    51    28  047E          MOVC3   R1,(R2),(R5)                   ;COPY VALUE TO INPUT BUFFER
                   54  8ED0  0482          POPL    R4                             ;RESTORE FLAGS
                   4D    11  0485          BRB     30$                            ;
                             0487
                             0487  95$:    STATUS  BUFOVF                         ;SET OVERFLOW STATUS
                 0120    30  048E          BSBW    DCL$LOCKED_STATE               ;RESTORE LOCKED KEYPAD STATE
                 00E0    31  0491          BRW     67$                            ;RETURN ERROR
                             0494
                             0494  96$:    STATUS  SYMOVF                         ;SET OVERFLOW STATUS
                 00D4    31  049B          BRW     66$                            ;RETURN
                             049E
                             049E  ;
                             049E  ; /NOERASE - COPY STRING INTO INPUT BUFFER.
                             049E  ;
      7E    0C AE      51    C1  049E  20$:    ADDL3   R1,12(SP),-(SP)                ;SAVE NEW LENGTH
    00000100 8F        6E    D1  04A3          CMPL    (SP),#WRK_C_INPBUFSIZ          ;WILL STRING FIT IN BUFFER?
                       DB    1A  04AA          BGTRU   95$                            ;NO, THEN RETURN ERROR
 55    0C AE      10 AE      C1  04AC          ADDL3   16(SP),12(SP),R5               ;FIND CURRENT EOL
             50      04 AE   D0  04B2          MOVL    4(SP),R0                       ;GET ADDRESS OF RAB
             53      0F A0   9A  04B6          MOVZBL  RAB$W_STV2+1(R0),R3            ;GET CURSOR OFFSET FROM EOL
                   55    53  C2  04BA          SUBL    R3,R5                          ;BACK UP TO INSERTION POINT
     7E       10 AE      53  C3  04BD          SUBL3   R3,16(SP),-(SP)                ;SAVE INITIAL OFFSET
                         3E  BB  04C2          PUSHR   #^M<R1,R2,R3,R4,R5>            ;SAVE REGISTERS
      6541      65       53  28  04C4          MOVC3   R3,(R5),(R5)[R1]               ;MOVE THE TEXT
                         3E  BA  04C9          POPR    #^M<R1,R2,R3,R4,R5>            ;RESTORE REGISTERS
             54   DD  04CB          PUSHL   R4                              ;SAVE FLAGS
  65   62    51   28  04CD          MOVC3   R1,(R2),(R5)                    ;COPY VALUE TO INPUT BUFFER
             54 8ED0  04D1          POPL    R4                              ;RESTORE FLAGS
                      04D4
                      04D4  ;
                      04D4  ; PROCESS /SET_STATE AND /LOCK
                      04D4  ;
          00DA    30  04D4  30$:    BSBW    DCL$LOCKED_STATE                ;RESTORE LOCKED KEYPAD STATE
       1C 54  02  E1  04D7          BBC     #SYM_V_STATE,R4,50$             ;BRANCH IF NO SET_STATE SPECIFIED
          52  51  D0  04DB          MOVL    R1,R2                           ;GET LENGTH/ADDR OF STATE STRING
          51  82  9A  04DE          MOVZBL  (R2)+,R1                        ;
         FB1C'    30  04E1          BSBW    DCL$ALLOC_STATE                 ;ALLOCATE IT
          AD 50   E9  04E4          BLBC    R0,96$                          ;BRANCH IF NO ROOM FOR SYMBOL
       0C 54  03  E1  04E7          BBC     #SYM_V_LOCK,R4,50$              ;BRANCH IF NOT /LOCK
       50  4C AB  D0  04EB          MOVL    PRC_L_LASTKEY(R11),R0           ;GET OLD LOCKED STATE
         FB0E'    30  04EF          BSBW    DCL$DEALLOC_STATE               ;DEALLOCATE IT
           48 AB  D0  04F2          MOVL    PRC_L_CURRKEY(R11),-            ;LOCK NEW KEY STATE
           4C AB      04F5                  PRC_L_LASTKEY(R11)              ;
                      04F7
                      04F7  ;
                      04F7  ; PROCESS /ERASE.
                      04F7  ;
       34 54  04  E1  04F7  50$:    BBC     #SYM_V_ERASE,R4,52$             ;IF /NOERASE, THEN SKIP
          FE81    30  04FB          BSBW    ERASE_LINE                      ;ERASE THE LINE
       2D 54  01  E1  04FE          BBC     #SYM_V_TERMINATE,R4,52$         ;IF /NOTERMINATE, THEN SKIP
       55  0C AB  D0  0502          MOVL    PRC_L_OUTRAB(R11),R5            ;SET ADDRESS OF OUTPUT FILE RAB
       50  10 AB  D0  0506  51$:    MOVL    PRC_L_TRMLIST(R11),R0           ;GET ADDRESS OF XABTRM ITEM LIST
           10 A0  D0  050A          MOVL    ITRM_L_PMPTADDR(R0),-           ;WRITE THE PROMPT STRING
           28 A5      050D                  RAB$L_RBF(R5)                   ;
           0C A0  B0  050F          MOVW    ITRM_W_PMPTLEN(R0),-            ;
           22 A5      0512                  RAB$W_RSZ(R5)                   ;
                      0514          DISABLE                                 ;DISABLE CONTROL Y/C AST'S
        06    00  F0  0518          INSV    #0,#RAB$V_PPF_RAT,-             ;DISABLE CR FORMAT WRITES
     02 A5    08      051B                  #RAB$S_PPF_RAT,RAB$W_ISI(R5)    ;
                      051E          $PUT    RAB=(R5)                        ;OUTPUT RECORD
        06    02  F0  0527          INSV    #FAB$M_CR,#RAB$V_PPF_RAT,-      ;ENABLE CR FORMAT WRITES
     02 A5    08      052A                  #RAB$S_PPF_RAT,RAB$W_ISI(R5)    ;
                      052D          ENABLE                                  ;ENABLE CONTROL Y/C AST'S
                      052F
                      052F  ;
                      052F  ; PROCESS /ECHO.
                      052F  ;
       4C 54  01  E1  052F  52$:    BBC     #SYM_V_TERMINATE,R4,60$         ;IF /NOTERMINATE, THEN SKIP
       38 54  00  E1  0533          BBC     #SYM_V_ECHO,R4,65$              ;IF /NOECHO, THEN SKIP
       55  0C AB  D0  0537          MOVL    PRC_L_OUTRAB(R11),R5            ;SET ADDRESS OF OUTPUT FILE RAB
       50  04 AE  D0  053B  55$:    MOVL    4(SP),R0                        ;GET SIZE OF OUTPUT RECORD
  51   10 AE  50  C1  053F          ADDL3   R0,16(SP),R1                    ;FIND END OF OUTPUT RECORD
          61  0D  90  0544          MOVB    #^X0D,(R1)                      ;INSERT CR AT END
          50  6E  C2  0547          SUBL    (SP),R0                         ;SUBTRACT OUT INITIAL OFFSET
  28 A5   51  50  C3  054A          SUBL3   R0,R1,RAB$L_RBF(R5)             ;SET ADDRESS OF OUTPUT RECORD
  22 A5   50  01  A1  054F          ADDW3   #1,R0,RAB$W_RSZ(R5)             ;SET SIZE OF OUTPUT RECORD
                      0554          DISABLE                                 ;DISABLE CONTROL Y/C AST'S
        06    00  F0  0558          INSV    #0,#RAB$V_PPF_RAT,-             ;DISABLE CR FORMAT WRITES
     02 A5    08      055B                  #RAB$S_PPF_RAT,RAB$W_ISI(R5)    ;
                      055E          $PUT    RAB=(R5)                        ;OUTPUT RECORD
        06    02  F0  0567          INSV    #FAB$M_CR,#RAB$V_PPF_RAT,-      ;ENABLE CR FORMAT WRITES
     02 A5    08      056A                  #RAB$S_PPF_RAT,RAB$W_ISI(R5)    ;
```

```
                              056D   859             ENABLE                                        ;ENABLE CONTROL Y/C AST'S
                              056F   860
            50    01    D0    056F   861  65$:       MOVL    #1,R0                                 ;SET NORMAL RETURN STATUS
                  8E    D5    0572   862  66$:       TSTL    (SP)+                                 ;POP INITIAL OFFSET
         0C AE    8E    D0    0574   863  67$:       MOVL    (SP)+,12(SP)                          ;REPLACE OLD LENGTH WITH NEW
            54    8E    7D    0578   864             MOVQ    (SP)+,R4                              ;RESTORE R4/R5
            52    8E    7D    057B   865             MOVQ    (SP)+,R2                              ;RESTORE R2/R3
                        05    057E   866             RSB                                           ;RETURN
                              057F   867
                              057F   868  ;
                              057F   869  ; PROCESS /NOTERMINATE.
                              057F   870  ;
         50    10 AB    D0    057F   871  60$:       MOVL    PRC_L_TRMLIST(R11),R0                 ;GET ADDRESS OF XABTRM ITEM LIST
      28 A0       8E    D0    0583   872             MOVL    (SP)+,ITRM_L_OFFSET(R0)               ;SET CHARACTER TO START ECHOING AT
               28 A0    D6    0587   873             INCL    ITRM_L_OFFSET(R0)                     ;
      03 54       04    E1    058A   874             BBC     #SYM_V_ERASE,R4,61$                   ;IF /NOERASE, THEN SKIP
               28 A0    D4    058E   875             CLRL    ITRM_L_OFFSET(R0)                     ;START WITH FIRST CHARACTER
      0C AE       8E    D0    0591   876  61$:       MOVL    (SP)+,12(SP)                          ;REPLACE OLD LENGTH WITH NEW
   18 A0       0C AE    B0    0595   877             MOVW    12(SP),ITRM_W_INILEN(R0)              ;SET LENGTH OF INITIAL STRING
   1C A0       08 AE    D0    059A   878             MOVL    8(SP),ITRM_L_INIADDR(R0)              ;SET ADDRESS OF INITIAL STRING
            54    8E    7D    059F   879             MOVQ    (SP)+,R4                              ;RESTORE R4/R5
         50    40 A4    D0    05A2   880             MOVL    RAB$L_XAB(R4),R0                      ;GET ADDRESS OF XABTRM
                  30    D0    05A6   881             MOVL    #ITRM_K_LENGTH,-                      ;SET EXPANDED LENGTH OF ITEM LIST
         0C A0                05A8   882                     XAB$W_ITMLST_LEN(R0)                  ;
            52    8E    7D    05AA   883             MOVQ    (SP)+,R2                              ;RESTORE R2/R3
            50    03    D0    05AD   884             MOVL    #3,R0                                 ;SET REPROMPT RETURN STATUS
                        05    05B0   885             RSB                                           ;RETURN
```

```
                      05B1             .SBTTL  RESTORE LOCKED KEYPAD STATE
                      05B1  ;---
                      05B1  ; DCL$LOCKED_STATE - RESTORE LOCKED KEYPAD STATE
                      05B1  ;
                      05B1  ; DELETE ANY TEMPORARY KEYPAD STATE THAT MAY HAVE BEEN SET AND RESTORE THE
                      05B1  ; LOCKED KEYPAD STATE.
                      05B1  ;
                      05B1  ; INPUTS:
                      05B1  ;
                      05B1  ;          R11 = ADDRESS OF PRC DATA STRUCTURE
                      05B1  ;          PRC_L_CURRKEY = CURRENT, POSSIBLY TERMPORARY, KEYPAD STATE
                      05B1  ;          PRC_L_LASTKEY = DEFAULT LOCKED KEYPAD STATE
                      05B1  ;
                      05B1  ; OUTPUTS:
                      05B1  ;
                      05B1  ;          PRC_L_CURRKEY IS UPDATED.
                      05B1  ;
                      05B1  ;---
                      05B1
                      05B1  DCL$LOCKED_STATE::                                    ;RESTORE LOCKED KEYPAD STATE
             50   DD  05B1             PUSHL   R0                                 ;SAVE R0
       48 AB      D1  05B3             CMPL    PRC_L_CURRKEY(R11),-               ;IS TEMPORARY STATE IN EFFECT?
       4C AB          05B6                     PRC_L_LASTKEY(R11)                 ;
          0C      13  05B8             BEQL    90$                                ;NO, THEN DONE
  50   48 AB      D0  05BA             MOVL    PRC_L_CURRKEY(R11),R0              ;GET ADDRESS OF ASCIC STATE
        FA3F'     30  05BE             BSBW    DCL$DEALLOC_STATE                  ;DEALLOCATE TEMPORARY STATE
       4C AB      D0  05C1             MOVL    PRC_L_LASTKEY(R11),-               ;RESTORE THE LOCKED STATE
       48 AB          05C4                     PRC_L_CURRKEY(R11)                 ;
          50    8ED0  05C6  90$:       POPL    R0                                 ;RESTORE R0
                  05  05C9             RSB                                        ;RETURN
                      05CA
```

```
                          05CA        .SBTTL  EXPAND INPUT LINE
                          05CA  ;---
                          05CA  ; EXPAND - EXPAND INPUT LINE WITH SYMBOL SUBSTITUTIONS
                          05CA  ;
                          05CA  ; SUBROUTINE TO EXPAND INPUT LINE BY EXECUTING ALL STRING SUBSTITUTION
                          05CA  ; COMMANDS.  THE UNUSED AREA IN THE EXPANSION BUFFER IS USED TEMPORARILY
                          05CA  ; TO HOLD THE EXPANDED COPY OF THE COMMAND LINE.  AFTER ALL SUBSTITUTIONS
                          05CA  ; ARE PERFORMED, THE EXPANDED COPY IS COPIED BACK INTO THE INPUT RECORD
                          05CA  ; BUFFER SO THAT SEMANTIC PARSING CAN CONTINUE.
                          05CA  ;
                          05CA  ; INPUTS:
                          05CA  ;
                          05CA  ;       WRK_L_CHARPTR = POINTER TO NEW INPUT RECORD
                          05CA  ;
                          05CA  ; OUTPUTS:
                          05CA  ;
                          05CA  ;       R2 = ADDRESS OF NEW INPUT RECORD
                          05CA  ;       R3 = LENGTH OF NEW INPUT RECORD, EXCLUDING NULL AT END OF LINE
                          05CA  ;       WRK_L_CHARPTR = POINTER TO EXPANDED INPUT RECORD
                          05CA  ;---
                          05CA
                          05CA  EXPAND:                                         ;EXPAND INPUT LINE
         03F0 8F     BB   05CA          PUSHR   #^M<R4,R5,R6,R7,R8,R9>          ;SAVE REGISTERS
         F48A CA     DD   05CE          PUSHL   WRK_L_MARKPTR(R10)              ;SAVE MARKER POINTER
         F486 CA     DD   05D2          PUSHL   WRK_L_EXPANDPTR(R10)            ;SAVE EXPANSION BUFFER POINTER
      7E   F0 AA     B0   05D6          MOVW    WRK_W_FLAGS(R10),-(SP)          ;SAVE CURRENT PARSING FLAGS
F0 AA    0C20 8F     A8   05DA          BISW    #WRK_M_INPSUBST!WRK_M_NOUPCASE!WRK_M_STAR,WRK_W_FLAGS(R10)
                          05E0                                                  ;PREVENT PROCESSING OF ',-,@,ETC.
                          05E0                                                  ;AND PREVENT UPCASING OF INPUT CHARS
                          05E0                                                  ;AND ACCEPT '*' AS A TERMINATOR
              7E     B4   05E0          CLRW    -(SP)                           ;INITIALIZE ITERATION COUNTER
                          05E2  ;
                          05E2  ; GET NEXT CHARACTER FROM INPUT RECORD
                          05E2  ;
           FA1B'     30   05E2  10$:    BSBW    DCL$GETCHAR                     ;GET NEXT CHARACTER FROM INPUT LINE
        27    50     91   05E5          CMPB    R0,#^A/'/                       ;STRING SUBSTITUTION COMMAND?
              15     12   05E8          BNEQ    70$                             ;IF NEQ NO
  27 F0 AA    04     E1   05EA          BBC     #WRK_V_QUOTE,WRK_W_FLAGS(R10),20$ ;IF LBC NOT IN QUOTE
           FA0E'     30   05EF          BSBW    DCL$SETCHAR                     ;CHECK NEXT CHARACTER
        27    50     91   05F2          CMPB    R0,#^A/'/                       ;NEXT CHARACTER ALSO SINGLE QUOTE?
              05     12   05F5          BNEQ    60$                             ;IF NEQ NO
           FA06'     30   05F7          BSBW    DCL$GETCHAR                     ;GOBBLE SECOND SINGLE QUOTE
              1A     11   05FA          BRB     20$                             ;TRY SYMBOL SUBSTITUTION
                          05FC  ;
                          05FC  ; ONE SINGLE QUOTE WAS DETECTED IN A DOUBLE QUOTED STRING - TREAT LITERALLY
                          05FC  ;
        50    27     9A   05FC  60$:    MOVZBL  #^A/'/,R0                       ;INSERT SINGLE QUOTE WITHIN STRING
                          05FF  ;
                          05FF  ; IF COMMENT IS DETECTED, THEN DO SUBSTITUTIONS BUT IF AN ERROR IS
                          05FF  ; DETECTED, THEN NO ERROR IS ISSUED AND A NULL STRING IS SUBSTITUTED.
                          05FF  ;
  0A F0 AA    04     E0   05FF  70$:    BBS     #WRK_V_QUOTE,WRK_W_FLAGS(R10),80$ ;BRANCH IF IN QUOTED STRING
        21    50     91   0604          CMPB    R0,#^A'!'                       ;START OF COMMENT STRING?
              05     12   0607          BNEQ    80$                             ;BRANCH IF NOT A COMMENT
                          0609          SETBIT  WRK_V_COMMENT,WRK_W_FLAGS(R10)  ;MARK WE ARE IN A COMMENT
                          060E                                                  ;SO THAT ERRORS ARE NOT REPORTED
                          060E  ;
```

```
                              060E  ; WRITE THE CURRENT CHARACTER INTO THE EXPANSION BUFFER AND LOOP
                              060E  ;
                 F9EF'   30   060E  80$:    BSBW    DCL$PUTCHAR                 ;PUT CHARACTER IN EXPANSION BUFFER
                    CF   12   0611          BNEQ    10$                         ;IF NOT EOL, KEEP SCANNING
                  00B9   31   0613          BRW     90$                         ;END OF LINE - TERMINATE SCAN
                              0616  ;
                              0616  ; SYMBOL SUBSTITUTION REQUESTED.  GET THE SYMBOL AND SEARCH THE SYMBOL
                              0616  ; TABLE, AND IF NOT FOUND THERE, TRY AS A LEXICAL FUNCTION.
                              0616  ;
             F486 CA     DD   0616  20$:    PUSHL   WRK_L_EXPANDPTR(R10)        ;SAVE PLACE IN EXPANSION BUFFER
         7E    F0 AA     B0   061A          MOVW    WRK_W_FLAGS(R10),-(SP)      ;SAVE LEXICAL FLAGS
      F0 AA       10     AA   061E          BICW    #WRK_M_QUOTE,WRK_W_FLAGS(R10)
                  0C     E0   0622          BBS     #WRK_V_COMMENT,-            ;ARE WE IN A COMMENT?
              06 F0 AA        0624                  WRK_Q_FLAGS(R10),22$
F0 AA    0800 8F         AA   0627          BICW    #WRK_M_NOUPCASE,WRK_W_FLAGS(R10)  ;NO, THEN UPCASE THE SYMBOL
                              062D                                              ;PRETEND WE'RE NOT IN A STRING
                              062D                                              ;SO GETOKEN STOPS BEFORE END-OF-STRING
            50    27     9A   062D  22$:    MOVZBL  #^A/'/,R0                   ;INSERT SINGLE QUOTE WITHIN STRING
               F9CD'     30   0630          BSBW    DCL$PUTCHAR                 ;IN CASE SUBSTITION NOT ALLOWED
               F9CA'     30   0633          BSBW    DCL$GETOKEN                 ;GET/COPY NEXT TOKEN
                              0636  ;
                              0636  ; IF IN COMMENT, ONLY ALLOW F$VERIFY TO BE SUBSTITUTED
                              0636  ;
    29 F0 AA      0C     E1   0636          BBC     #WRK_V_COMMENT,WRK_W_FLAGS(R10),28$ ;BRANCH IF NOT IN COMMENT
          04      51     D1   063B          CMPL    R1,#4                       ;AT LEAST 4 CHARACTER TOKEN?
                  1C     19   063E          BLSS    25$                         ;IF NOT, SKIP SYMBOL SUBSTITUTION
          7E      62     D0   0640          MOVL    (R2),-(SP)                  ;PUSH FIRST FOUR CHARACTERS
          7E      51     7D   0643          MOVQ    R1,-(SP)                    ;SAVE DESCRIPTOR
          51      04     9A   0646          MOVZBL  #4,R1                       ;CREATE TEMPORARY DESCRIPTOR
    52         08 AE     9E   0649          MOVAB   8(SP),R2
               F9B0'     30   064D          BSBW    DCL$UPCASE                  ;UPCASE THE SYMBOL
          51      8E     7D   0650          MOVQ    (SP)+,R1                    ;RESTORE THE REGISTERS
45562446 8F       8E     D1   0653          CMPL    (SP)+,#^A'F$VE'             ;IS IT F$VERIFY WITHIN A COMMENT?
                  08     13   065A          BEQL    28$                         ;IF SO, ALLOW SUBSTITUTION
   02 AE     F486 CA     D0   065C  25$:    MOVL    WRK_L_EXPANDPTR(R10),2(SP)  ;COPY 'SYMBOL TO EXPANSION BUFFER
                  45     11   0662          BRB     50$
                  51     D5   0664  28$:    TSTL    R1                          ;ZERO LENGTH SYMBOL?
                  1F     13   0666          BEQL    40$                         ;IF EQL YES
          56      51     7D   0668          MOVQ    R1,R6                       ;SAVE STRING PARAMETERS
               F992'     30   066B          BSBW    DCL$SEARCH                  ;SEARCH FOR SYMBOL
               16 50     E8   066E          BLBS    R0,40$                      ;IF LBS SYMBOL DEFINITION FOUND
                              0671          CLRBIT  WRK_V_INPSUBST,WRK_W_FLAGS(R10) ;DO PROCESSING OF !,-,@,ETC.
                              0676                                              ;TO ALLOW CONTINUATIONS IN FUNCT. ARGS
                  0C     E1   0676          BBC     #WRK_V_COMMENT,-            ;ARE WE IN A COMMENT?
            09 F0 AA          0678                  WRK_Q_FLAGS(R10),32$
          51      56     7D   067B          MOVQ    R6,R1                       ;YES, THEN UPCASE "F$VERIFY"
               F97F'     30   067E          BSBW    DCL$UPCASE
          56      51     7D   0681          MOVQ    R1,R6
               F979'     30   0684  32$:    BSBW    DCL$LEXIF                   ;EVALUATE LEXICAL FUNCTION
               F976'     30   0687  40$:    BSBW    DCL$CVT_STRING              ;CONVERT RESULT TO CHARACTER STRING
                              068A          SETBIT  WRK_V_INPSUBST,WRK_W_FLAGS(R10) ;DISABLE !,-,@,ETC.
                              068F  ;
                              068F  ; TRAILING SINGLE QUOTES ARE OPTIONAL AFTER SYMBOL - GOBBLE IT
                              068F  ;
               F96E'     30   068F          BSBW    DCL$SETCHAR                 ;PEEK AT NEXT CHARACTER
          27      50     91   0692          CMPB    R0,#^A/'/                   ;SYMBOL END WITH SINGLE QUOTE?
                  03     12   0695          BNEQ    30$                         ;BRANCH IF TRAILING SINGLE QUOTE
             F966'  30  0697       BSBW    DCL$GETCHAR              ;GOBBLE TRAILING QUOTE
                        069A ;
                        069A ; APPEND THE SYMBOL TRANSLATION TO THE FRONT OF THE INPUT BUFFER
                        069A ; AND RESET THE INPUT POINTER TO POINT TO IT.  THIS IS DONE IN CASE
                        069A ; THERE ARE ANY SINGLE QUOTES IN THE TRANSLATION WHICH REQUIRE SUBSTITUTION.
                        069A ;
       F48E CA   51     C2  069A 30$:   SUBL    R1,WRK_L_CHARPTR(R10)    ;CALCULATE ADDRESS TO COPY STRING
       50   F48E CA     D0  069F        MOVL    WRK_L_CHARPTR(R10),R0    ;POINT TO NEW POSITION IN INBUF
    01 A0   62   51     28  06A4        MOVC    R1,(R2),1(R0)            ;CONCATENATE STRING TO INPUT BUFFER
                            06A9 ;
                            06A9 ; A MAXIMUM OF 1000 SUBSTITUTIONS IS ALLOWED PER LINE, TO PREVENT ANY
                            06A9 ; INFINITE LOOPS FROM OCCURRING DUE TO RECURSIVE SUBSTITUTIONS.
                            06A9 ;
         F0 AA   8E     B0  06A9 50$:   MOVW    (SP)+,WRK_W_FLAGS(R10)   ;RESTORE FLAGS QUOTE AND NOUPCASE
            F486 CA   8ED0  06AD        POPL    WRK_L_EXPANDPTR(R10)     ;RETRIEVE ADDRESS IN EXPANSION BUFFER
FF28 6E   01   03E8 8F  3D  06B2        ACBW    #1000,#1,(SP),10$        ;CHECK FOR SUBSTITUTION LOOP
                            06BA        STATUS  EXPSYN                   ;EXPRESSION SYNTAX ERROR
     F486 CA   57   56  C1  06C1        ADDL3   R6,R7,WRK_L_EXPANDPTR(R10) ;POINT AT END OF SYMBOL
       F48A CA      77  9E  06C7        MOVAB   -(R7),WRK_L_MARKPTR(R10) ;SET ADDRESS OF """
               F931'    31  06CC        BRW     DCL$PARSERR              ;REPORT ERROR
                            06CF ;
                            06CF ; END OF LINE DETECTED.  MOVE EXPANDED LINE BACK INTO THE INPUT BUFFER
                            06CF ; FOR THE LEXICAL PROCESSING.
                            06CF ;
         57    04 AE    D0  06CF 90$:   MOVL    4(SP),R7                 ;GET SAVED EXPANSION POINTER
    59   F486 CA   57   C3  06D3        SUBL3   R7,WRK_L_EXPANDPTR(R10),R9 ;CALCULATE LENGTH OF EXPANDED LINE
         58   F996 CA   9E  06D9        MOVAB   WRK_G_INPBUF+WRK_C_INPBUFSIZ(R10),R8 ;FIND END OF INPUT BUFFER
              58   59   C2  06DE        SUBL    R9,R8                    ;COMPUTE ADDRESS TO MOVE LINE TO
         68   67   59   28  06E1        MOVC    R9,(R7),(R8)             ;MOVE EXPANDED LINE TO END OF INPUT BUFFER
              52   58   7D  06E5        MOVQ    R8,R2                    ;SET INPUT LINE PARAMETERS
                   53   D7  06E8        DECL    R3                       ;DECREMENT LENGTH TO EXCLUDE EOL CHAR
      F48E CA   FF A2   9E  06EA        MOVAB   -1(R2),WRK_L_CHARPTR(R10);SET ADDRESS OF EXPANDED INPUT LINE
                   8E   B5  06F0        TSTW    (SP)+                    ;REMOVE ITERATION COUNTER FROM STACK
         F0 AA     8E   B0  06F2        MOVW    (SP)+,WRK_W_FLAGS(R10)   ;RESTORE CURRENT PARSING FLAGS
            F486 CA   8ED0  06F6        POPL    WRK_L_EXPANDPTR(R10)     ;RESTORE EXPANSION BUFFER POINTER
            F48A CA   8ED0  06FB        POPL    WRK_L_MARKPTR(R10)       ;RESTORE MARKER POINTER
            03F0 8F     BA  0700        POPR    #^M<R4,R5,R6,R7,R8,R9>   ;RESTORE REGISTERS
                        05  0704        RSB
```

```
                     0705               .SBTTL  SPECIAL TOKEN LEXICAL PROCESSING
                     0705       ;---
                     0705       ;
                     0705       ; THIS ROUTINE IS CALLED TO PROCESS LEXICAL TOKENS WITH SPECIAL CHARACTERS.
                     0705       ; THE LIST OF CHARACTERS ARE:
                     0705       ;
                     0705       ;       &SYMBOL         - TOKEN IS SUBSTITUTED WITH SYMBOL VALUE
                     0705       ;       @FILESPEC       - TOKEN IS SUBSTITUTED WITH FIRST RECORD CONTAINED
                     0705       ;                         IN THE ASSOCIATED PROCEDURE FILE.
                     0705       ;
                     0705       ; INPUTS:
                     0705       ;
                     0705       ;       R11 = ADDRESS OF PRC AREA
                     0705       ;       R10 = ADDRESS OF WRK AREA
                     0705       ;       R0 = FIRST CHARACTER IN TOKEN
                     0705       ;
                     0705       ; ADDITIONAL INPUT AND OUTPUT SPECIFICATIONS ARE GIVEN WITH EACH ROUTINE.
                     0705       ;
                     0705       ;---
                     0705
                     0705       DCL$SPECIAL::
      26    50   91  0705               CMPB    R0,#^A'&'                       ;SUBSTITUTION?
            07   13  0708               BEQL    AMPERSAND                       ;BRANCH IF SO
   40 8F    50   91  070A               CMPB    R0,#^A'@'                       ;INDIRECTION?
            22   13  070E               BEQL    INDIRECT                        ;BRANCH IF SO
                 05  0710               RSB                                     ;IF NOT KNOWN, IGNORE IT
```

```
                        0711  1099          .SBTTL  PROCESS &SYMBOL CONSTRUCT
                        0711  1100  ;---
                        0711  1101  ;
                        0711  1102  ; HANDLE &SYMBOL CONSTRUCT.
                        0711  1103  ;
                        0711  1104  ; INPUTS:
                        0711  1105  ;
                        0711  1106  ;       R0 = CHARACTER IN INPUT BUFFER (&)
                        0711  1107  ;       R1/R2 = DESCRIPTOR OF TOKEN, INCLUDING "&" CHARACTER
                        0711  1108  ;
                        0711  1109  ; OUTPUTS:
                        0711  1110  ;
                        0711  1111  ;       R0 = NEXT CHARACTER IN INPUT BUFFER
                        0711  1112  ;       R1/R2 = DESCRIPTOR OF UPDATED TOKEN
                        0711  1113  ;
                        0711  1114  ;       THE EXPANSION BUFFER WILL BE OVERWRITTEN WITH SYMBOL VALUE
                        0711  1115  ;
                        0711  1116  ;---
                        0711  1117  AMPERSAND:
                3C  BB  0711  1118          PUSHR   #^M<R2,R3,R4,R5>        ;SAVE REGISTERS
                52  D6  0713  1119          INCL    R2                      ;POINT TO SYMBOL NAME
                51  D7  0715  1120          DECL    R1                      ;ADJUST LENGTH OF SYMBOL NAME
                03  13  0717  1121          BEQL    40$                     ;BRANCH IF NULL SYMBOL NAME
             F8E4'  30  0719  1122          BSBW    DCL$SYM_STRING          ;SEARCH LOCAL/GLOBAL SYMBOL TABLES
     00 BE  62  51  28  071C  1123  40$:    MOVC    R1,(R2),@(SP)           ;OVERWRITE "&SYMBOL" WITH ITS VALUE
       F486 CA  53  D0  0721  1124          MOVL    R3,WRK_L_EXPANDPTR(R10) ;SET NEW EXPANSION BUFFER POINTER
                3C  BA  0726  1125          POPR    #^M<R2,R3,R4,R5>        ;RESTORE REGISTERS
             F8D5'  30  0728  1126          BSBW    DCL$SETCHAR             ;PEEK AT NEXT CHARACTER IN INPUT BUFFER
51     F486 CA  52  C3  072B  1127          SUBL3   R2,WRK_L_EXPANDPTR(R10),R1 ;CALCULATE LENGTH OF NEW TOKEN
                    05  0731  1128          RSB
```

```
                              0732   1130            .SBTTL  PROCESS @FILESPEC CONSTRUCT
                              0732   1131    ;---
                              0732   1132    ;
                              0732   1133    ; HANDLE @FILESPEC CONSTRUCT.
                              0732   1134    ;
                              0732   1135    ; INPUTS:
                              0732   1136    ;
                              0732   1137    ;       R0 = CHARACTER IN INPUT BUFFER (@)
                              0732   1138    ;
                              0732   1139    ;       AT THIS POINT, ONLY THE @ HAS BEEN SEEN, AND THE FILESPEC
                              0732   1140    ;       HAS NOT YET BEEN PROCESSED.
                              0732   1141    ;
                              0732   1142    ; OUTPUTS:
                              0732   1143    ;
                              0732   1144    ;       R0 = NEXT CHARACTER IN INPUT BUFFER
                              0732   1145    ;---
                              0732   1146    INDIRECT:
     19 68 AB    05     E2    0732   1147            BBSS    #PRC_V_IND,PRC_W_FLAGS(R11),90$ ;SKIP IF INDIRECTION DISABLED
                              0737   1148                                            ;AND DISABLE WHILE PROCESSING FILESPEC
         F48A CA        DD    0737   1149            PUSHL   WRK_L_MARKPTR(R10)      ;SAVE OLD PARSE POSITION
             F8C2'      30    073B   1150            BSBW    DCL$MARK                ;SET PARSE POSITION FOR '@' PROCESSING
             F8BF'      30    073E   1151            BSBW    DCL$STACKIND            ;STACK CURRENT INDIRECT LEVEL
                              0741   1152            CLRBIT  PRC_V_IND,PRC_W_FLAGS(R11) ;ENABLE INDIRECTION AGAIN
           09 50        E9    0745   1153            BLBC    R0,DCL$CHARERROR        ;BRANCH IF ERROR DETECTED
    F48A CA      8E     D0    0748   1154            MOVL    (SP)+,WRK_L_MARKPTR(R10) ;RESTORE OLD PARSE POSITION
             F8B5       30    074D   1155            BSBW    DCL$INPUT               ;GET FIRST LINE, CHARACTER OF PROCEDURE
                        05    0750   1156    90$:    RSB
```

```
                 0751         .SBTTL  ERROR HANDLER IN CHARACTER INPUT ROUTINES
                 0751  ;---
                 0751  ;
                 0751  ; THIS ROUTINE IS CALLED TO PERFORM ANY SPECIAL PROCESSING WHEN AN
                 0751  ; ERROR IS DETECTED BY THE CHARACTER INPUT ROUTINES.
                 0751  ;
                 0751  ; INPUTS:
                 0751  ;
                 0751  ;       R0 = STATUS CODE
                 0751  ;
                 0751  ; OUTPUTS:
                 0751  ;
                 0751  ;       NONE
                 0751  ;---
                 0751  DCL$CHARERROR::
F8AC'   30       0751          BSBW    DCL$ERRORMSG            ;REPORT ERROR MESSAGE
   50   DD       0754          PUSHL   R0                      ;SAVE STATUS CODE
F8A7'   30       0756          BSBW    DCL$FLUSH               ;FLUSH INPUT BUFFER
   50 8ED0       0759          POPL    R0                      ;RESTORE STATUS CODE
F8A1'   30       075C          BSBW    DCL$SET_STATUS          ;SET COMPLETION STATUS
F89E'   31       075F          BRW     DCL$RESTART             ;START THE PARSING ALL OVER AGAIN
```

```
                     0762          .SBTTL RECALL COMMAND
                     0762  ;+
                     0762  ; DCL$RECALL - RECALL COMMAND
                     0762  ;
                     0762  ; THIS ROUTINE IS CALLED TO EXECUTE THE DCL RECALL COMMAND.  THE RECALL
                     0762  ; COMMAND REPROMPTS THE USER WITH A COMMAND THAT HE HAS PREVIOUSLY ENTERED
                     0762  ; OR DISPLAYS FOR THE USER THE LIST OF ALL THE COMMANDS IN THE COMMAND BUFFER.
                     0762  ;
                     0762  ; INPUTS:
                     0762  ;
                     0762  ;       R8 = ADDRESS OF SCRATCH BUFFER DESCRIPTOR
                     0762  ;       R9 = ADDRESS OF SCRATCH STACK
                     0762  ;       R10 = ADDRESS OF COMMAND WORK AREA
                     0762  ;       R11 = ADDRESS OF PROCESS WORK AREA
                     0762  ;
                     0762  ; OUTPUTS:
                     0762  ;
                     0762  ;       R0 = STATUS CODE
                     0762  ;-
                     0762
                     0762  DCL$RECALL::
                     0762
                     0762  ;
                     0762  ; SKIP IF ENTERED FROM A COMMAND PROCEDURE.
                     0762  ;
         06     E0   0762          BBS     #PRC_V_MODE,-               ; IF SET, NOT INTERACTIVE
   0A 68 AB          0764                  PRC_Q_FLAGS(R11),3$         ;
         0B     E0   0767          BBS     #PRC_V_YLEVEL,-             ; IF SET, AT CONTROL Y/C LEVEL
   08 68 AB          0769                  PRC_Q_FLAGS(R11),5$         ;
      5C AB     D5   076C          TSTL    PRC_L_INDEPTH(R11)          ; INDIRECT LEVEL ZERO?
         03     13   076F          BEQL    5$                          ; BRANCH IF YES
       0113     31   0771  3$:     BRW     90$                         ; RETURN
                     0774
                     0774
                     0774  ;
                     0774  ; REMOVE THIS COMMAND FROM THE RECALL BUFFER.
                     0774  ;
    012F CB     D0   0774  5$:     MOVL    PRC_L_RECALLPTR(R11),-      ; UPDATE WRK RECALL PTR
      EA AA          0778                  WRK_L_RECALLPTR(R10)        ;
         7E     7C   077A          CLRQ    -(SP)                       ; ALLOCATE COMMAND DESCRIPTOR
         5E     DD   077C          PUSHL   SP                          ; POINT TO IT
 0000'CF 01     FB   077E          CALLS   #1,DCL$GET_PREV_COMMAND     ; GET THE RECALL COMMAND
         8E     7C   0783          CLRQ    (SP)+                       ; DISCARD THE DESCRIPTOR
      EA AA     D0   0785          MOVL    WRK_L_RECALLPTR(R10),-      ; UPDATE PRC RECALL PTR
    012F CB          0788                  PRC_L_RECALLPTR(R11)        ;
                     078B
                     078B  ;
                     078B  ; PARSE THE COMMAND.
                     078B  ;
   56    01     D0   078B          MOVL    #1,R6                       ; ASSUME BACKING UP ONE COMMAND
      F86F'     30   078E          BSBW    DCL$GETDVAL                 ; GET NEXT DESCRIPTOR
   55    04     D1   0791          CMPL    #PTR_K_ENDLINE,R5           ; EOL?
         6C     13   0794          BEQL    30$                         ; YES, EXECUTE THE COMMAND
   55    03     D1   0796          CMPL    #PTR_K_PARAMETR,R5          ; NUMBER?
         24     12   0799          BNEQ    20$                         ; NO, MUST BE /ALL
   52    51     7D   079B          MOVQ    R1,R2                       ; COPY DESCRIPTOR
   7E    52     7D   079E          MOVQ    R2,-(SP)                    ; SAVE IT ON THE STACK
```

```
                  51      01  D0  07A1          MOVL    #1,R1                     ; SET DECIMAL RADIX
                       F859'  30  07A4          BSBW    DCL$CNVNOEDIT             ; CONVERT NUMBER TO BINARY
                  52      8E  7D  07A7          MOVQ    (SP)+,R2                  ; RESTORE THE DESCRIPTOR
                          50  D5  07AA          TSTL    R0                        ; WAS IT A NUMBER?
                          05  13  07AC          BEQL    10$                       ; YES, CHECK BOUNDS
                  56      52  7D  07AE          MOVQ    R2,R6                     ; SAVE THE STRING DESCRIPTOR
                          19  11  07B1          BRB     35$                       ; GET THE COMMAND
                  14      51  D1  07B3  10$:    CMPL    R1,#WRK_C_RECALLMAX       ; WITHIN BOUNDS?
                          0C  1A  07B6          BGTRU   95$                       ; NO, SIGNAL ERROR
                  56      51  D0  07B8          MOVL    R1,R6                     ; GET BACKUP COUNT
                          07  13  07BB          BEQL    95$                       ; SIGNAL ERROR IF ZERO
                          43  11  07BD          BRB     30$                       ; GET THE COMMAND
                  56      14  D0  07BF  20$:    MOVL    #WRK_C_RECALLMAX,R6       ; /ALL WAS SPECIFIED
                          5F  11  07C2          BRB     50$                       ; DISPALY THE COMMANDS
                                  07C4
                                  07C4  95$:    STATUS  IVVALU                    ; SET INVALID VALUE STATUS
                              05  07CB          RSB                               ; RETURN
                                  07CC
                                  07CC  ;
                                  07CC  ; FETCH SPECIFIED COMMAND BY STRING.
                                  07CC  ;
                  59      14  D0  07CC  35$:    MOVL    #WRK_C_RECALLMAX,R9       ; /ALL WAS SPECIFIED
                          7E  7C  07CF          CLRQ    -(SP)                     ; ALLOCATE COMMAND DESCRIPTOR
                          5E  DD  07D1  36$:    PUSHL   SP                        ; PUSH DESCR ADDRESS
             0000'CF      01  FB  07D3          CALLS   #1,DCL$GET_PREV_COMMAND   ; GET THE PREVIOUS COMMAND
       04 B8      04 BE   6E  28  07D8          MOVC3   (SP),@4(SP),@4(R8)        ; COPY COMMAND TO SCRATCH BUFFER
                  68      6E  D0  07DE          MOVL    (SP),(R8)                 ; COPY INITIAL LENGTH
                  51      68  7D  07E1          MOVQ    (R8),R1                   ; SET COMMAND DESCRIPTOR
                       F819'  30  07E4          BSBW    DCL$TRIM                  ; UPCASE AND TRIM THE COMMAND
                  51      56  D1  07E7          CMPL    R6,R1                     ; MAKE SURE WE ARE CHECKING A SUBSTRING
                          08  1A  07EA          BGTRU   361$                      ; SKIP THIS ONE IF NOT
   62    56    00   67    56  2D  07EC          CMPC5   R6,(R7),#0,R6,(R2)        ; DO THE STRINGS MATCH?
                          1A  13  07F2          BEQL    37$                       ; YES, THEN REPROMPT
                       DA 59  F5  07F4  361$:   SOBGTR  R9,36$                    ; LOOP TILL SEARCHED ALL COMMANDS
                       5E 08  C0  07F7          ADDL    #8,SP                     ; RESTORE THE STACK
          50   00038238 8F    D0  07FA          MOVL    #CLI$_CMDNOTFND,R0        ; SET COMMAND NOT FOUND STATUS
                              05  0801          RSB                               ; RETURN
                                  0802
                                  0802  ;
                                  0802  ; FETCH SPECIFIED COMMAND BY NUMBER.
                                  0802  ;
                          7E  7C  0802  30$:    CLRQ    -(SP)                     ; ALLOCATE COMMAND DESCRIPTOR
                          5E  DD  0804  32$:    PUSHL   SP                        ; PUSH DESCR ADDRESS
             0000'CF      01  FB  0806          CALLS   #1,DCL$GET_PREV_COMMAND   ; GET THE PREVIOUS COMMAND
                       F6 56  F5  080B          SOBGTR  R6,32$                    ; LOOP TILL BACKED UP FAR ENOUGH
                                  080E
                                  080E  ;
                                  080E  ; SET UP RAB TO REPROMPT WITH THE COMMAND.
                                  080E  ;
                  51      8E  7D  080E  37$:    MOVQ    (SP)+,R1                  ; GET COMMAND DESCRIPTOR
              54     14 AB    D0  0811          MOVL    PRC_L_INDINPRAB(R11),R4   ; ASSUME USING INDIRECT INPUT RAB
                          0B  E1  0815          BBC     #PRC_V_YLEVEL,-           ; BRANCH IF NOT AT CTRL/Y LEVEL
                    04 68 AB      0817                  PRC_Q_FLAGS(R11),40$      ;
              54     08 AB    D0  081A          MOVL    PRC_L_INPRAB(R11),R4      ; USE LEVEL 0 INPUT RAB
                        FBCE  30  081E  40$:    BSBW    INSERT_COMMAND            ; INSERT THE COMMAND
                          64  11  0821          BRB     90$                       ;
                                  0823
```

```
                         0823  ;
                         0823  ; DISPLAY ALL COMMANDS
                         0823  ;
   20312000 8F      D0   0823  50$:    MOVL    #^X20312000,-             ; SET INITIAL NUMBER STRING
       F890 CA           0829                  WRK_G_INPBUF-6(R10)       ;
  50     EA AA      D0   082C  55$:    MOVL    WRK_L_RECALLPTR(R10),R0   ; HAVE WE ALREADY DISPLAYED THEM ALL?
  50     FF A0      9E   0830          MOVAB   -1(R0),R0                 ;
 51    0133 CB      9E   0834          MOVAB   PRC_G_COMMANDS(R11),R1    ; WRAP IF NECESSARY
       51   50      D1   0839          CMPL    R0,R1                     ;
            05      1E   083C          BGEQU   57$                       ;
 50    0533 CB      9E   083E          MOVAB   PRC_G_COMMANDS+PRC_C_CMDBUFSIZ-1(R11),R0
            60      95   0843  57$:    TSTB    (R0)                      ; BRANCH IF NO COMMANDS LEFT
            40      13   0845          BEQL    90$                       ;
            7E      7C   0847          CLRQ    -(SP)                     ; ALLOCATE COMMAND DESCRIPTOR
            5E      DD   0849          PUSHL   SP                        ; PUSH DESCR ADDRESS
 0000'CF    01      FB   084B          CALLS   #1,DCL$GET_PREV_COMMAND   ; GET THE PREVIOUS COMMAND
       51   8E      7D   0850          MOVQ    (SP)+,R1                  ; GET COMMAND DESCRIPTOR
       51   03      C0   0853          ADDL    #3,R1                     ; INSERT THE COMMAND NUMBER
       52   03      C2   0856          SUBL    #3,R2                     ;
       7E   51      7D   0859          MOVQ    R1,-(SP)                  ; SAVE COMMAND DESCRIPTOR
         F7A1'      30   085C          BSBW    DCL$MSGOUT                ; OUTPUT THE COMMAND
       51   8E      7D   085F          MOVQ    (SP)+,R1                  ; RESTORE COMMAND DESCRIPTOR
  39     01 A2      91   0862          CMPB    1(R2),#^X39               ; IS ONES DIGIT A 9?
            11      12   0866          BNEQ    60$                       ; NO, THEN SKIP
       31   62      91   0868          CMPB    (R2),#^X31                ; IS TENS DIGIT A 1?
            07      12   086B          BNEQ    58$                       ; NO, THEN SKIP
 62    2F32 8F      B0   086D          MOVW    #^X2F32,(R2)              ; YES, INSERT "2/"
            05      11   0872          BRB     60$                       ; BRANCH
 62    2F31 8F      B0   0874  58$:    MOVW    #^X2F31,(R2)              ; INSERT "1/"
         01 A2      96   0879  60$:    INCB    1(R2)                     ; INCREMENT THE COMMAND NUMBER
                         087C
                         087C  ;
                         087C  ; CHECK FOR NO MORE COMMANDS.
                         087C  ;
       012F CB      D1   087C          CMPL    PRC_L_RECALLPTR(R11),-    ; HAVE WE RETURNED TO THE ORIGIN?
         EA AA           0880                  WRK_L_RECALLPTR(R10)      ;
            03      13   0882          BEQL    90$                       ;
         A5 56      F5   0884          SOBGTR  R6,55$                    ; OR DISPLAYED THE MAX # OF COMMANDS?
                         0887
                         0887  90$:    STATUS  NORMAL                    ; SET SUCCESS
                    05   088E          RSB                               ; ALL DONE
                         088F
                         088F          .END
```

```
$$.TMP1                     = 00000001                   ERASE_LINE                    0000037F  R    02
$$.TMP2                     = 00000065                   ERRORT                        000000AE  R    02
$$T1                        = 00000001                   EXE$C_SYSEFN                  *********   X  02
ABORT                         0000028D  R    02          EXPAND                        000005CA  R    02
AMPERSAND                     00000711  R    02          FAB$M_CR                    = 00000002
CLI$_BUFOVF                 = 00038018                   FAB$W_IFI                   = 00000002
CLI$_CMDNOTFND              = 00038238                   GET_INPUT                     00000045  R    02
CLI$_EXPSYN                 = 00038038                   GET_KEY_NAME                  *********   X  02
CLI$_IVVALU                 = 00038088                   INDIRECT                      00000732  R    02
CLI$_NORMAL                 = 00030001                   INSERT_COMMAND                000003EF  R    02
CLI$_SYMOVF                 = 00038138                   IO_ERROR                      0000019B  R    02
CL'$_USGOTO                 = 00038148                   ITRM_C_LENGTH                 00000030
DCL$ABORT                     *********   X  02          ITRM_C_MINLEN                 00000018
DCL$ALLOC_STATE               *********   X  02          ITRM_K_LENGTH                 00000030
DCL$BACKUPCHAR                *********   X  02          ITRM_K_MINLEN                 00000018
DCL$CHARERROR                 00000751  RG   02          ITRM_L_INIADDR                0000001C
DCL$CLOSE_PPFS                *********   X  02          ITRM_L_INIRET                 00000020
DCL$CNVNOEDIT                 *********   X  02          ITRM_L_MODIFIERS              00000004
DCL$CRLF                      *********   X  02          ITRM_L_MODRET                 00000008
DCL$CVT_STRING                *********   X  02          ITRM_L_OFFRET                 0000002C
DCL$DEALGOTO                  *********   X  02          ITRM_L_OFFSET                 00000028
DCL$DEALLOC_STATE             *********   X  02          ITRM_L_PMPTADDR               00000010
DCL$DISABLE                   *********   X  02          ITRM_L_PMPTRET                00000014
DCL$DSBCONTRLY                *********   X  02          ITRM_W_INICODE                0000001A
DCL$ERRORMSG                  *********   X  02          ITRM_W_INILEN                 00000018
DCL$FLUSH                     *********   X  02          ITRM_W_MODCODE                00000002
DCL$GETCHAR                   *********   X  02          ITRM_W_MODLEN                 00000000
DCL$GETDVAL                   *********   X  02          ITRM_W_OFFCODE                00000026
DCL$GETOKEN                   *********   X  02          ITRM_W_OFFLEN                 00000024
DCL$GET_CURR_COMMAND          *********   X  02          ITRM_W_PMPTCODE               0000000E
DCL$GET_NEXT_COMMAND          *********   X  02          ITRM_W_PMPTLEN                0000000C
DCL$GET_PREV_COMMAND          *********   X  02          PRC_B_CONTINUE                000000F3
DCL$INPUT                     00000005  RG   02          PRC_B_DEFRADIX                000000AE
DCL$LEXIF                     *********   X  02          PRC_B_EXMDEPMOD               000000AD
DCL$LOCKED_STATE              000005B1  RG   02          PRC_B_EXMDEPWID               000000AC
DCL$MARK                      *********   X  02          PRC_B_EXONLYL                 0000012D
DCL$MSGOUT                    *********   X  02          PRC_B_FLAGS2                  000000AF
DCL$PARSERR                   *********   X  02          PRC_B_IMGFLAG                 00000078
DCL$PUTCHAR                   *********   X  02          PRC_B_OUTFLAGS                0000012C
DCL$PUT_COMMAND               *********   X  02          PRC_B_PROMPTLEN               000000F0
DCL$PUT_SEGMENT               *********   X  02          PRC_C_CMDBUFSIZ             = 00000401
DCL$RECALL                    00000762  RG   02          PRC_C_LENGTH                  00000534
DCL$RESTART                   *********   X  02          PRC_G_COMMANDS                00000133
DCL$SEARCH                    *********   X  02          PRC_G_PROMPT                  000000F4
DCL$SEARCH_KEYPAD             *********   X  02          PRC_K_LENGTH                  00000534
DCL$SETCHAR                   *********   X  02          PRC_L_CURRKEY                 00000048
DCL$SET_STATUS                *********   X  02          PRC_L_EXMDEPADR               000000A8
DCL$SPECIAL                   00000705  RG   02          PRC_L_EXTARG                  00000094
DCL$STACKIND                  *********   X  02          PRC_L_EXTBLK                  0000008C
DCL$SYM_STRING                *********   X  02          PRC_L_EXTCOD                  0000009C
DCL$TRIM                      *********   X  02          PRC_L_EXTHND                  00000090
DCL$UNSTACK                   *********   X  02          PRC_L_EXTPRM                  00000098
DCL$UPCASE                    *********   X  02          PRC_L_IDFLNK                  000000BC
DEV$V_TRM                     *********   X  02          PRC_L_IMGACTSTS               00000080
DVI$_DEVDEPEND2             = 0000001C                   PRC_L_INDCLOCK                0000007C
END_OF_LIST                   00000322  R    02          PRC_L_INDEPTH                 0000005C
ERASE                         00000000  R    02          PRC_L_INDFAB                  0000001C
```

```
PRC_L_INDINPRAB                  00000014          PRC_W_PMPTCTRL                   000000F1
PRC_L_INDOUTRAB                  00000018          PRC_W_WAITIOSB                   00000066
PRC_L_INPRAB                     00000008          PROCESS_ESCAPE                   00000407 R     02
PRC_L_LASTKEY                    0000004C          PROCESS_INPUT                    000000B1 R     02
PRC_L_LSTSTATUS                  000000B0          PROCESS_RECALL                   000002C5 R     02
PRC_L_ONCTLY                     000000B8          PTR_B_LEVEL                      00000004
PRC_L_ONERROR                    0000006C          PTR_B_NUMBER                     00000005
PRC_L_OUTOFBAND                  000000B4          PTR_B_PARMCNT                    00000006
PRC_L_OUTRAB                     0000000C          PTR_B_VALUE                      00000000
PRC_L_OUTRABCTX                  00000118          PTR_C_LENGTH                     0000000C
PRC_L_PPFLIST                    00000070          PTR_K_ENDLINE                  = 00000004
PRC_L_RECALLPTR                  0000012F          PTR_K_LENGTH                     0000000C
PRC_L_RESTART                    00000058          PTR_K_PARAMETR                 = 00000003
PRC_L_SAVAP                      00000000          PTR_L_DESCR                      00000000
PRC_L_SAVFP                      00000004          PTR_L_ENTITY                     00000008
PRC_L_SEVERITY                   00000050          RAB$L_CTX                      = 00000018
PRC_L_SPWN                       000000C0          RAB$L_RBF                      = 00000028
PRC_L_STACKLM                    000000A4          RAB$L_STS                      = 00000008
PRC_L_STACKPT                    000000A0          RAB$L_STV                      = 0000000C
PRC_L_STATUS                     00000054          RAB$L_UBF                      = 00000024
PRC_L_STS                        00000084          RAB$L_XAB                      = 00000040
PRC_L_STV                        00000088          RAB$S_PPF_RAT                  = 00000008
PRC_L_SYMBOL                     00000060          RAB$V_PPF_IND                  = 0000000E
PRC_L_TMBX                       00000074          RAB$V_PPF_RAT                  = 00000006
PRC_L_TRMLIST                    00000010          RAB$W_ISI                      = 00000002
PRC_Q_ALLOCREG                   00000020          RAB$W_RSZ                      = 00000022
PRC_Q_COMMAND                    000000E0          RAB$W_STV0                     = 0000000C
PRC_Q_FLUSHTIME                  000000D0          RAB$W_STV2                     = 0000000E
PRC_Q_GLOBAL                     00000028          RAB$W_USZ                      = 00000020
PRC_Q_IMAGENAME                  000000D8          RECALL_CURR                      00000315 R     02
PRC_Q_KEYPAD                     00000040          RECALL_NEXT                      0000032E R     02
PRC_Q_LABEL                      00000030          RECALL_PREV                      000002CF R     02
PRC_Q_LOCAL                      00000038          REINP                            00000007 R     02
PRC_Q_SAVEPRIV                   000000E8          RETURN                           0000017F R     02
PRC_T_OUTDVI                     0000011C          RMS$_CONTROLY                    ********   X   02
PRC_V_AUTOLOGO                 = 00000008          RMS$_EOF                         ********   X   02
PRC_V_CARRCNTL                 = 00000000          RMS$_RSA                         ********   X   02
PRC_V_CNTRLY                   = 00000001          RMS$_SYS                         ********   X   02
PRC_V_EOFLOGO                  = 0000000E          SILENT_LOGOUT                    0000029B RG    02
PRC_V_FLUSH                    = 00000006          SPECIAL                          00000196 R     02
PRC_V_GOTO                     = 00000004          SS$_EXQUOTA                      ********   X   02
PRC_V_IND                      = 00000005          STATUS                           00000287 R     02
PRC_V_MODE                     = 00000006          SYM_B_FLAGS                      0000000B
PRC_V_VERIFY                   = 00000007          SYM_B_NONUNIQUE                  0000000B
PRC_V_YLEVEL                   = 0000000B          SYM_B_TYPE                       0000000A
PRC_W_ASTIOSB                    000000C6          SYM_L_BL                         00000004
PRC_W_ASTRETN                    000000C8          SYM_L_FL                         00000000
PRC_W_ASTSTATUS                  000000C4          SYM_T_SYMBOL                     0000000C
PRC_W_ATTMBX                     0000007A          SYM_V_ECHO                     = 00000000
PRC_W_FLAGS                      00000068          SYM_V_ERASE                    = 00000004
PRC_W_INPCHAN                    00000064          SYM_V_LOCK                     = 00000003
PRC_W_ONLEVEL                    0000006A          SYM_V_STATE                    = 00000002
PRC_W_OUTIFI                     00000114          SYM_V_TERMINATE                = 00000001
PRC_W_OUTISI                     00000116          SYM_W_SIZE                       00000008
PRC_W_OUTMBXCHN                  000000CA          SYS$CANCEL                       ********  GX   02
PRC_W_OUTMBXREF                  000000CE          SYS$CANEXH                       ********  GX   02
PRC_W_OUTMBXSIZ                  000000CC          SYS$CLOSE                        ********  GX   02
```

```
SYS$EXIT                        ********  GX    02          WRK_W_PMPTLEN                        FFFFF99E
SYS$GET                         ********  GX    02          XAB$W_ITMLST_LEN                   = 0000000C
SYS$GETDVIW                     ********  GX    02          _$$_                               = 000000EF
SYS$PUT                         ********  GX    02
SYS$WAIT                        ********  GX    02
TT2$V_ANSICRT                 = 00000018
WRK_B_CMDOPT                    FFFFFFC3
WRK_B_MAXPARM                   FFFFFFD0
WRK_B_MINPARM                   FFFFFFD1
WRK_B_PARMCNT                   FFFFFFCE
WRK_B_PARMSUM                   FFFFFFCF
WRK_B_RECALLCNT                 FFFFFFC5
WRK_B_VALLEV                    FFFFFFC4
WRK_B_VERBTYP                   FFFFFFC2
WRK_C_INPBUFSIZ               = 00000100
WRK_C_LENGTH                    FFFFF486
WRK_C_RECALLMAX               = 00000014
WRK_G_BUFFER                    FFFFF492
WRK_G_INPBUF                    FFFFF896
WRK_G_RESULT                    FFFFF9B6
WRK_K_LENGTH                    FFFFF486
WRK_L_CHARPTR                   FFFFF48E
WRK_L_DISALLOW                  FFFFFFE6
WRK_L_ERRORRTN                  FFFFF9AE
WRK_L_EXPANDPTR                 FFFFF486
WRK_L_IMAGE                     FFFFFFE2
WRK_L_MARKPTR                   FFFFF48A
WRK_L_PAROUT                    FFFFFFD2
WRK_L_PMPTADDR                  FFFFF9A2
WRK_L_PROMPTRTN                 FFFFF9A6
WRK_L_PROPTR                    FFFFFFC6
WRK_L_QUABLK                    FFFFFFCA
WRK_L_READRTN                   FFFFF9AA
WRK_L_RECALLPTR                 FFFFFFEA
WRK_L_RSLEND                    FFFFFFB6
WRK_L_RSLNXT                    FFFFFFBA
WRK_L_SAVAP                     FFFFFFF8
WRK_L_SAVFP                     FFFFFFFC
WRK_L_SAVSP                     FFFFFFF4
WRK_L_SIGNALRTN                 FFFFFFD6
WRK_L_SPECRTN                   FFFFF9B2
WRK_L_TAB_VEC                   FFFFFFDE
WRK_L_VERB                      FFFFFFBE
WRK_M_INPSUBST                = 00000400
WRK_M_NOUPCASE                = 00000800
WRK_M_QUOTE                   = 00000010
WRK_M_STAR                    = 00000020
WRK_V_COMMAND                 = 00000001
WRK_V_COMMENT                 = 0000000C
WRK_V_CONTIN                  = 00000003
WRK_V_INPSUBST                = 0000000A
WRK_V_INQUIRE                 = 00000007
WRK_V_QUOTE                   = 00000004
WRK_V_TRAILSPC                = 00000009
WRK_W_FLAGS                     FFFFFFF0
WRK_W_FLAGS2                    FFFFFFF2
WRK_W_IMGCHAN                   FFFFFFEE
```

```
                                                        +----------------+
                                                        ! Psect synopsis !
                                                        +----------------+

PSECT name                       Allocation          PSECT No.  Attributes
----------                       ----------          ---------  ----------
.  ABS  .                        00000000  (     0.)  00 (  0.)  NOPIC     USR     CON     ABS     LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                            FFFFFFFC  (     0.)  01 (  1.)  NOPIC     USR     CON     ABS     LCL NOSHR   EXE   RD     WRT NOVEC BYTE
DCL$ZCODE                        0000088F  (  2191.)  02 (  2.)  NOPIC     USR     CON     REL     LCL NOSHR   EXE   RD   NOWRT NOVEC BYTE

                                                     +------------------------+
                                                     ! Performance indicators !
                                                     +------------------------+

Phase                      Page faults    CPU Time       Elapsed Time
-----                      -----------    --------       ------------
Initialization                       9    00:00:00.07    00:00:00.80
Command processing                  80    00:00:00.70    00:00:06.75
Pass 1                             351    00:00:15.31    00:00:43.74
Symbol table sort                    0    00:00:01.60    00:00:04.80
Pass 2                             235    00:00:03.98    00:00:10.02
Symbol table output                 33    00:00:00.23    00:00:00.64
Psect synopsis output                2    00:00:00.03    00:00:00.26
Cross-reference output               0    00:00:00.00    00:00:00.00
Assembler run totals               710    00:00:21.94    00:01:07.02

The working set limit was 1500 pages.
79799 bytes (156 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1072 non-local and 94 local symbols.
1336 source lines were read in Pass 1, producing 22 object records in Pass 2.
58 pages of virtual memory were used to define 41 macros.

                                                     +--------------------------+
                                                     ! Macro library statistics !
                                                     +--------------------------+

Macro library name                             Macros defined
------------------                             --------------
_$255$DUA28:[SYSLIB]SYSBLDMLB.MLB;1                   0
_$255$DUA28:[DCL.OBJ]DCL.MLB;1                       13
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                        0
_$255$DUA28:[SYSLIB]STARLET.MLB;2                    20
TOTALS (all libraries)                               33

1301 GETS were required to define 33 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:READREC/OBJ=OBJ$:READREC MSRC$:READREC/UPDATE=(ENH$:READREC)+EXECML$/LIB+LIB$:DCL/LIB+SYS$LIBRARY:SYSBLDMLB/LIB
```

0072 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY